AFRL-HE-WP-TP-2007-0007

A Prototype Laboratory for Developing Human Performance Representation Interchange Standards

Nils LaVine

Micro Analysis & Design, Inc.
4949 Pearl East Circle, Suite 300
Boulder CO 80301-2477

March 2003

Interim Report for May 2002 – March 2003

Air Force Research Laboratory
Human Effectiveness Directorate
Warfighter Interface Division
Cognitive Systems Branch
Wright-Patterson AFB OH 45433-7604

NOTICE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the Air Force Research Laboratory, Det 1, Wright Site, Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-HE-WP-TP-2007-0007

HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE DIRECTOR

//SIGNED//

DANIEL G. GODDARD
Chief, Warfighter Interface Division
Human Effectiveness Directorate
Air Force Research Laboratory

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)*	2. REPORT TYPE	3. DATES COVERED *(From - To)*
March 2003	Interim	May 2002 - March 2003

4. TITLE AND SUBTITLE
A Prototype Laboratory for Developing Human Performance Representation Interchange Standards

5a. CONTRACT NUMBER
N61339-02-C-0114

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
63832D

6. AUTHOR(S)
Nils LaVine

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER
0476DM00

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

Micro Analysis & Design, Inc.
4949 Pearl East Circle, Suite 300
Boulder CO 80301-2477

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Air Force Materiel Command
Air Force Research Laboratory
Human Effectiveness Directorate
Warfighter Interface Division
Cognitive Systems Branch
Wright-Patterson AFB OH 45433-7604

Defense Modeling and Simulation Office
1901 N. Beauregard Street, Suite 500
Alexandria VA 22311-1705

10. SPONSOR/MONITOR'S ACRONYM(S)
AFRL/HECS, DMSO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
AFRL-HE-WP-TP-2007-0007

12. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution is unlimited.
AFRL/PA cleared on 27 June 2007, AFRL-WS-07-1531.

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This report summarizes the objectives, technologies involved, technical approach, and results of an effort to provide improved behaviors to dismounted soldiers performing an operation of clearing a building in a constructive simulation. The ultimate objective was to improve upon behavioral representation within entity based simulations by employing a client-server architecture that included discrete event simulations, cognitive models, and a Computer Generated Force application for Dismounted Infantry (DI) operations in a Military Operations in Urban Terrain (MOUT) environment. The goals of this effort were focused on being able to modify the Computer Generated Force (CGF) application to utilize information provided by a behavioral server and also to improve upon the behavioral representation capabilities within the CGF application. Situational awareness information provided to a behavioral server allows for more complex behaviors than those available in the configuration managed version of the CGF application. This program was successful in all of these aspects.

15. SUBJECT TERMS Human Behavior Modeling, Client-Server Architecture, Computer Generated Forces (CGF), Constructive Simulations, Military Operations in Urban Terrain (MOUT)

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			John L. Camp
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	SAR	64	19b. TELEPHONE NUMBER *(include area code)*

Standard Form 298 (Rev. 8-98)
Prescribed by ANSI Std. 239.18

THIS PAGE LEFT INTENTIONALLY BLANK

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

THIS PAGE LEFT INTENTIONALLY BLANK

1.0 Introduction

1.1 Program Overview

This technical effort was sponsored by the Defense Modeling and Simulation Office (DMSO) and performed over a ten-month period under the Naval Air Warfare Center contract N61339-02-C-0114, "A Prototype Laboratory for Developing Human Performance Representation Interchange Standards." This program will be referred to as the Human Behavior Representation Laboratory (HBR Lab) program for the remainder of this report.

This report summarizes the objectives, technologies involved, technical approach, and results of this program. The change in focus over the course of the program will be discussed along with other programmatic issues.

The underlying objective of this effort was to provide improved behaviors to dismounted soldiers performing an operation of clearing a building in a constructive simulation. However, the ultimate objective was to improve upon behavioral representation within entity based simulations by employing a client-server architecture that included discrete event simulations, cognitive models, and a Computer Generated Force application for Dismounted Infantry (DI) operations in a Military Operations in Urban Terrain (MOUT) environment. The goals of this effort were focused on being able to modify the Computer Generated Force (CGF) application to not only utilize information provided by a behavioral server, but to also further improve upon the behavioral representation capabilities within the CGF application. Situational awareness information provided to a behavioral server will allow for more complex behaviors than those available in the configuration managed version of the CGF application. This program was successful in all of these aspects.

1.2 Report overview

This report about the HBR Lab program provides the following:

- background and programmatic issues that shaped this effort,
- technologies involved with the successful accomplishment of this effort,
- description of our client-server architecture,
- discussion of the MOUT behaviors for clearing a building that were implemented within the constructive simulation,
- description of our HBR model of the implemented MOUT behaviors, and
- discussion of future work and recommendations for; functional improvements to other MOUT behaviors, the selected constructive simulation, and the client-server architecture.

First, we will discuss the direction and focus of this program. Next, in Section 2 we will introduce the technologies involved in realizing this effort. That will be followed in Section 3 by a discussion of the implementation of each of the technologies. Finally, we will provide the program results including lessons learned, results and conclusions.

Appendix A is also included that contains a self contained model built in Micro Saint of a squad clearing a building. Its purpose is to model and understand actual human "gut level" decisions

that occur while clearing a building, without being constrained by modeling or fidelity limitations imposed by the CGF.

1.3 Issues Impacting the HBR Lab Program

This program received direction and focus including the desire to provide a useful and relevant product to the Modeling & Simulation (M&S) community. Included are factors which will each be discusses in the sub-sections that follow:

- Improved realism of CGF behavior representation
- A change in emphasis for the program
- Modeling military operations in urban terrain (MOUT)
- Software integration into a CGF of interest

1.3.1 CGF Behavioral Representation Realism

It is widely acknowledged that behavioral representation within CGF systems is inadequate. In study after study of CGFs, the lack of credible behavioral representation is consistently identified as a severe limiting feature. CGF entities tend to behave in a rote, non-human, and predictable manner. This issue alone can severely discredit results, thereby defeating the basic intention of the CGF system. CGFs' purpose of providing battlefield entities with realistic behaviors is not being met and is significantly reducing their effectiveness and usefulness. This includes how behavior influences actions such as battlefield stressors, the actual performance of humans within a system, and how decisions made by humans are modeled for CGF entities. The inadequate representation of realistic behaviors has implications in many, if not all, of CGFs intended purposes (e.g., training, concept evaluation, and test and evaluation.)

This lack of credible behavioral representation in CGFs was a key driver in the focus of this program. In this HBR Lab program, our primary goal was to improve upon the utility of CGF applications by improving entity behavioral representation. This will be demonstrated in an example using the clear building behavior and will be presented as a methodology for improved behavior representation in general by presenting the methodology used and the client-server approach.

1.3.2 Change in Focus of the HBR Lab Program

The original focus of the HBR Lab program was described as: "Design, integrate, and document a laboratory capable of running a modest collection of (i.e., several) CGFs from DoD and academia, and NPCs from EI and academia that exchange data and control according to the evolving HBR interchange standard. The laboratory shall also be capable of coordinating and interoperating with one or more similarly funded and chartered laboratories. It is expected that the capabilities of the laboratory will scale, over time, to meet the needs of larger and more complex exercises." (NPCs are Non-Player Characters, and EI is the Entertainment Industry)

Initially we worked in this direction; however it quickly became apparent that many of the disparate HBR Labs that were sponsored by DMSO had varying level of maturity and capability. DMSO leadership quickly realized that this would limit the utility of other HBR Labs or induce undue pressure on some of the maturing capabilities. As a result, the HBR Labs structure was "de-linked" and DMSO managed each of the DMSO HBR laboratories separately.

As a result of the "de-linking" of laboratories, our effort was refocused on improving behavioral representation within CGF applications performing MOUT on the synthetic battlefield.

1.3.3 Military Operations In Urban Terrain Modeling & Simulation

MOUT and DI operations are domains that are of increasing interest to the military and M&S communities. The importance is due to the acknowledgement by the Defense community that MOUT will be the rule as opposed to the exception in combat of the future. Therefore, having the best M&S capabilities for MOUT is critical.

This HBR Lab program addresses the shortcomings of using CGF applications to examine Tactics, Techniques, & Procedures (TTPs) and advanced technologies to assist in MOUT. As the military services continue to transform, advanced capabilities must be evaluated prior to their implementation. The intent of this program is to improve the ability to investigate and study MOUT by providing more accurate representations of human behaviors in entity level simulations.

As leverage for this HBR Lab program we extended and enhanced the DMSO sponsored "Improved Behavioral Representation for Operations Other Than War Within Aggregate Level Simulations" program recently completed in May 2002. That program implemented a client-server architecture for incorporating improved representation into existing behaviors within a CGF application. In the HBR Lab program we use task network and cognitive models to provide improved fidelity behavioral representation to CGF entities. A requirement resulting from the ability to provide increased fidelity is to interchange Situational Awareness (SA) information between the CGF application and the behavioral models. This capability existed in a somewhat rudimentary form in earlier programs but was enhanced for these purposes. Also, the level of decisions and behaviors implemented in this program progressed beyond inflexible direct integration, or "plug in", to the existing CGF application as was done in the earlier program. To address the fidelity increase required for this clear building behavior, a substantial number of additional behaviors were added to the CGF application, a process not done in the earlier effort. The result is a dramatic increase in the fidelity level of behaviors and control of CGF entities for MOUT.

1.3.4 Obtaining CGF Applications for the HBR Lab Effort

This effort leverages work from a previous Operations Other Than War (OOTW) project and has a strong focus on MOUT behaviors. We therefore initially decided to utilize the available CGF application baseline of Dismounted Infantry Semi-Automated Forces (DISAF) version 7.1 with its core Distributed Interactive Simulation (DIS) communication protocol functionality. Part way into the project, DISAF version 9.4 and OneSAF Testbed Baseline – Joint Virtual Battlespace (OTB-JVB), which contains DISAF version 9.4 functionality, were identified as potentially better CGF application candidates for our intended effort. These CGF applications have enhanced entity capabilities compared with a now two-year old DISAF version 7.1. Also, the M&S community would be more appreciative and receptive of enhanced functionality in a more recent software baseline than DISAF version 7.1.

A significant amount of time was spent in procuring both DISAF version 9.4 and OTB-JVB. We examined each line of the software code, to determine the viability of integrating our DISAF version 7.1 enhancements. Unfortunately, the late reception of these CGF applications had a detrimental impact upon our ability to integrate our DISAF version 7.1 modifications. It quickly

became apparent that DISAF version 9.4 was extremely unstable and with the amount of time remaining in the program, we were unable to get it ready for delivery. Thus, it was eliminated as a candidate however with sufficient time it could easily be used.

OTB-JVB has many of the DISAF version 9.4 enhancements integrated into it and proved to be a much more stable code base. This baseline has the ability to operate in both DIS and HLA environments. The baseline had diverged significantly in its core network communications capabilities, but we were able to successfully integrate our DISAF version 7.1 enhancements, however only when using the DIS protocol. Because the HLA implementation deviated significantly from the implementation performed in the CGF application JointSAF (where we have implemented our architecture on a previous program), we again were without enough time to complete the HLA integration. We believe with additional time and resources we could have accomplished our integration of improved DI MOUT entity behaviors in OTB-JVB with HLA capabilities.

In summary, we have incorporated high-fidelity HBR, consisting of a human performance model and a cognitive model, into both DISAF v7.1 and OTB-JVB utilizing the DIS protocol. For the remainder of this report, when we refer to "SAF" or "SAFs", it refers to DISAF version 7.1 and the OTB-JVB baseline with DIS functionality only.

1.4 Knowledge Acquisition

Given that we were creating a new behavior for use in the Semi-Automated Force (SAF), it was important that we start with a validated baseline. To that end, a number of current Field Manuals (FMs) dealing with urban warfare were obtained from the US Army. Of specific interest was information pertaining to a) how infantry enter a building, and b) the actions taken to pass through and secure a building. The most informative manual in these areas was FM 3-06.11 "Combined Arms Operations in Urban Terrain". Several other FMs provided corroborating and supportive information: FM 7-8 "Infantry Rifle Platoon and Squad", TC90-1 "Training for Urban Operations", and FM 7-10 "Urban Operations".

2.0 Technologies Involved

This HBR Lab program leveraged technologies developed on previous programs, expanded their capabilities, and integrated their features. Technologies used include:

- Naturalistic Decision Making
- Task Network Modeling
- Server: HBR Server
- Client: Semi-Automated Forces
- Client-server architecture

Using these technologies, behavior models placed external to the SAF entity simulation provide the capability for both higher fidelity models, ease user access to modify the behavior, and provide selectable level of entity model fidelity. The result is a system that not only allows a SAF entity to be more accurately modeled, but also provides a mechanism for future enhancements to many other systems and behaviors.

A technology used for this program is Naturalist Decision Making. NDM has been evolving over the past 15 or so years but has only recently been implemented in a form usable for computational needs. For this effort we will be using Recognition Primed Decision (RPD) as an implementation of NDM. RPD will be used to make decisions for the model exercised in Micro Saint, a discrete event simulation tool.

Task Network Modeling (TNM) can be used as the external modeling application and is an ideal environment for developing models that accurately represent human behavior. One such tool for realizing TNM is Micro Saint, a Micro Analysis & Design product. It was developed and has been used specifically for human behavior representation and was used here to model tasks and decisions.

A capability with a specific function of providing, or serving, HBR to other simulations can be implemented with a HBR Server. High fidelity models of selected behaviors would reside in the server and would be available for use by other applications requesting that capability.

SAF packages provide entity representation on a synthetic battlefield. However, currently there are significant limitations with the modeling of the behaviors these entities perform. One method often attempted to satisfy the need for improved behaviors is to directly modify SAF code, a highly time consuming and costly development process. Often, the result of simply inserting higher fidelity, more complex models into an already large and cumbersome system like a SAF is reduced processing capabilities and an inefficient development effort. Each entity requires more and more computer processing time to evaluate the added algorithm thereby reducing the number of entities that can be supported. In addition, the process to add or modify a behavior may take many months to implement and requires a computer programmer adept at the SAF architecture and code. Often, the behavior model of entities is needed at a higher level of sophistication than is currently available. Additionally, there is often a desire to selectively choose specific high fidelity models for some entities and not for others.

What is desired is to provide the needed improved behavior representation in a more timely, efficient and robust manner. The client-server architecture and associated tools, and the associated task network modeling and corresponding behavior models, provide a greatly increased capability for improved human behavior representation. All of this becomes available in a timely manner (on the order of hours, days, or at most weeks) with very minimal intervention from a SAF programmer. Furthermore, these models, residing in an external server, can model high fidelity behaviors while still maintaining the same number of entities simulated. That is, while there is tremendous increase in the models of behaviors, there is no decrease in performance. The client-server architecture provides the capability to link entity behavior to entities in the synthetic environment of distributed simulations with variable fidelity human performance and system models. This approach places human/system models external to the SAF entity simulation. By having these behavior models external to the SAF and developed using a rapid development-modeling tool, extremely sophisticated models can be developed, changes can be made quickly, and their effects portrayed immediately within SAF. The client-server architecture provides a standardized capability for these external models to obtain situational awareness information about the entity and provide behavior attributes back to the SAF entities. The human/system models "serve" behavioral attributes to SAF entities using Distributed Interactive Simulation (DIS) or High Level Architecture (HLA) communications protocols for inter-model communication.

The tasks required to accomplish this objective of this program were the following:

- select a CGF,
- use an architecture that supports high fidelity, easily modifiable (composable) models of individual combatants performing building clearing operations,
- select behaviors involved with clearing a building that can benefit from improved behavioral representation,
- build task network models of behavioral of dismounted infantry performing operations to clear a building,
- create RPD models of human decisions of clearing a building using appropriate situational awareness cues,
- demonstrate the improved behaviors within the SAF, and
- document the SAF modifications, architecture, and behavior models.

2.1 Naturalistic Decision Making (NDM)

2.1.1 Background

Naturalistic decision making is a school of thought within cognitive psychology that takes the study of decision-making outside the controlled environment of the laboratory and "into the wild" where decisions are made under uncertain conditions, with incomplete information, severe time pressure and dramatic consequences. Rather than prescribe normative models of decision making, naturalistic models attempt to describe how people use their experience to arrive at good (not necessarily optimal) decisions. For example, Klein's model of the Recognition-Primed Decision (RPD) describes how people use their experience to "size up" a situation to form a sense of "typicality" (Klein, 1993; Klein, 1998; Klein, Calderwood, & Clinton-Cirocco, 1985). Typicality amounts to the recognition of goals, cues, expectancies and a course of action. Where traditional models of decision making postulate an analytical agent who carefully considers a host of alternatives, often against a background of perfect information, the RPD model postulates an agent poised to act who depends on his expertise to assess the available information and identify the first workable alternative. Figure 2-1 shows the flow of activities in the RPD model.

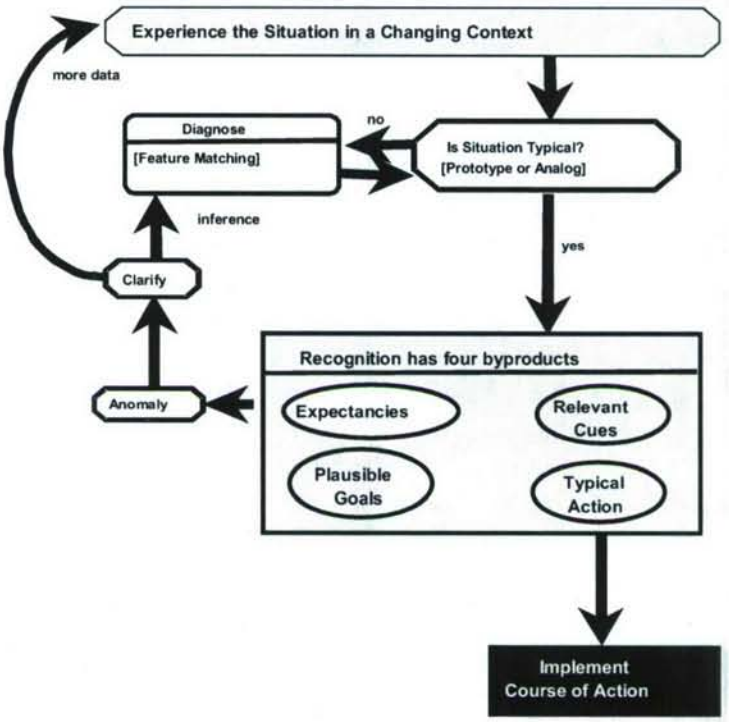

Figure 2-1. The RPD model (diagnostic variation).

There are several noteworthy contrasts between the RPD model and traditional models of decision-making. First, the RPD model emphasizes situation assessment over the comparison of options. The idea is that a suitable course of action will emerge once the situation is understood. Hence, the focus of decision-making in the RPD model shifts away from the kinds of activities often assumed to drive decision-making behavior (e.g., multi-attribute analysis) and onto "higher-level" perceptual activities and diagnosis. Second, the RPD model engenders a view of expertise as a product of experience rather than the elaboration of first-principles or model-based reasoning; that is, RPD depends on strong knowledge rather than weak, domain-independent knowledge. Third, the RPD model postulates a decision-maker who implements the first workable course of action he considers. If several alternatives emerge, they are considered serially, but more often than not, the first course of action to emerge is workable. Because a workable solution need not be an optimal solution, RPD is considered to be a satisfying model of decision making. Finally, RPD is a descriptive model of decision making; it does not dictate how decisions ought to be made and, by the same token, it does not specify the lower-level cognitive processes that are necessary to realize the flow of activities depicted in Figure 2-1.

2.1.2 Recognition Primed Decision (RPD) Model tool

Focusing our attention on Klein's descriptive model of the recognition-primed decision, a framework has been implemented that allows the human performance modeler to use more than just probabilistic draws or simple IF-THEN rules to represent human decision making. We emphasize recognition, experience and, more recently, prediction and diagnosis in the decision making process. The result is a set of data structures and routines that support a variety of user-

specified effects on the recognition process. For example, the user can specify, among other things, what cues the decision maker will attend to in a given situation, how much "weight" the decision maker will attach to each cue, how closely the decision maker will look at a given cue, how precisely the current situation will be recognized and how the decision maker's understanding of simple causal relationships might be used diagnostically to confirm an initial assessment of a situation.

The representation of naturalistic decision making integrated into the RPD model tool is built around an *episodic* model of long-term memory (Hintzman, 1984, 1986a, 1986b). Long-term memory contains a collection of decision-making episodes, each of which records the symbolic cues that prompted the recognition of the situation, the course of action that was taken, and a simple measure of the outcome of that course of action (success or failure). We model decisions one-at-a-time, so even if a single agent makes a variety of decision, we represent each decision separately. Our model uses a similarity-based recognition routine to compare a new decision making situation to all of those "remembered" situations. The product of recognition is a composite recollection of what to do based on the contribution that each past experience makes in proportion to its similarity to the current situation. In these respects, our naturalistic model of decision making is not unlike a rule-based production system; each episode could be thought of as an IF-THEN rule relating an antecedent condition (i.e., a set of cues) with an action and, likewise, the similarity-based recognition is reminiscent of the fuzzy pattern matching routines used in many production systems (e.g., Anderson & Lebiere, 1998). But, despite these seeming similarities to a production system, there are several respects in which our model of naturalistic decision making departs from a simple collection of IF-THEN rules.

First, in representing a decision, we allow the user to specify how much importance the decision maker will attach to each cue. The intuition here is that a given piece of information might be more or less significant depending on the situation. A cue that is highly diagnostic in one situation—e.g., an enemy marking on a distant vehicle—might be less diagnostic in another situation—e.g., when the decision maker receives fire from that vehicle. Likewise, one combination of cues might be more useful in assessing a situation than another combination, again, depending on the situation. By allowing the user to assign "weights" to each cue situation-by-situation (including, perhaps, a weight of zero), we provide a mechanism for representing a decision maker's dynamic sense of significance.

Second, we allow the user to specify how precisely the cues themselves will be discriminated. Although our decision maker is limited to a symbolic representation of cues that prompt recognition, the user can specify how finely a given cue will be "perceived." Consider the example of a cue that represents the distance to a target. We make no attempt to model the perceptual mechanisms that allow the decision maker to transform the two-dimensional sensory information impinging the retina into a meaningful three-dimensional judgment of distance. Rather, we simply assume that the decision maker will be directly aware of distance, perhaps as meters to target, and we allow the user to choose whether to represent the cue in terms of *near* and *far*, or *near*, *medium*, and *far*, or *very-near*, *near*, *medium*, *far*, *very-far,* etc. Moreover, in the same way we allow the weight attached to a cue to vary across situations, we also allow the resolution for a given cue to shift across situations. In some situations a coarse, ballpark estimate might do, while in others it might be necessary to take a closer look.

Third, we allow the user to specify a variety of parameters that directly affect the recognition process. One of these parameters, which can vary from situation to situation, controls how precise the decision maker will be in his recollection of past experiences. Another parameter, fixed in advance by the user, dictates how "certain" the decision maker must be in his recognition of the situation. Finally, we have recently implemented a causal Bayesian framework (Pearl, 1988, 2000) in which the user can represent the decision-maker's capacity for predictive and diagnostic reasoning. Here too there are a variety of parameters available to the user that dictate how much uncertainty the decision maker will tolerate in his decision making process.

In addition to the explicit representation of the process effects enumerated above, our data representation supports an implicit representation of uncertainty by allowing partial matches between cues. Returning to the example of distance to target, by specifying discrete ranges for each cue, the user casts the cue in subjective terms. A continuous cue is thus "interpreted" by the decision maker and, moreover, is subject to some misinterpretation. That is, our encoding scheme ensures that "adjacent" values like *very-far* and *far* are more similar than the values of *very-far* and *not-so-far*, while the values of *very-far* and *near* are completely dissimilar. Thus, the recognition of a situation in which a target is perceived to be *very-far* away will be influenced by experiences in long-term memory in which the target was only *far* away. Although this kind of misinterpretation is a natural consequence of our recognition routine, a similar kind of misinterpretation can be "induced" and thus serve as a proxy for a variety of workload effects. For instance, we allow the user to specify both how a decision maker will sample cues in his environment under both "normal" and "overloaded" conditions and how long a sampled cue will persist in working memory. Again, because the recognition routine allows partial matches, if a cue is missing form working memory—either because it hasn't been sampled, or has not been sampled recently enough—the recognition of the situation will be influenced by a wider variety of experiences and the decision maker's recall will be less precise.

The episodic model of long-term memory provides a theoretically satisfying recognition routine that supports a variety of process-level effects. The same model also provides a natural mechanism for incorporating the impact of experience on decision making. Each time the model makes a decision, that experience—the cues that prompted recognition, the course of action that was taken, and the outcome of that course of action—is recorded and will, in turn, influence the recognition of subsequent situations (this feature can be disabled by the user). This capability allows the model to adjust to its decision making environment. Moreover, the user specifies how outcomes are determined. In this way, the user can control how decision making behavior is reinforced. At one extreme, the user can dictate sets of conditions to be evaluated prior to or simultaneously with the decision (this yields essentially rule-based behavior); at the other extreme, the user can specify sets of conditions to be evaluated some fixed amount of time after the decision has been made (this yields behavior that relates dynamically to the environment). In either case, both the quantity and quality of experience can have an observable impact on the decisions being made.

We have made this wide variety of "naturalistic" functionality available to the user by way of a suite of graphical user interfaces. These interfaces allow the user to input the requisite information about the decision—the cues that prompt recognition, the available courses of action, the conditions by which outcomes are evaluated and specifications for the process-level effects described above. (There is also an option to describe the decision maker's diagnostic

reasoning and its concomitant feedback on recognition). This information is then used at run-time to create a long-term memory structure and to affect the decision making. In addition, long-term memory traces can be saved in files and analyzed off-line. Using another complementary suite of tools, the user can quickly determine how well decisions were made during the course of a previous simulation run (or runs); the user can also see how a hypothetical situation will be recognized given a particular long-term memory file; finally, the user can edit a long-term memory file either adding or deleting particular experiences to that file. These tools can be bundled in a Micro Saint task-network environment (as described below), they can be used in a stand-alone mode and linked to another Windows-based simulation environment via Microsoft's Component Object Model (COM) functionality, and we have experimented with embedding them directly into a CGF as an additional "decision server editor" within a tool bar to effect specific entity-level behaviors via a client-server architecture (again, as described below).

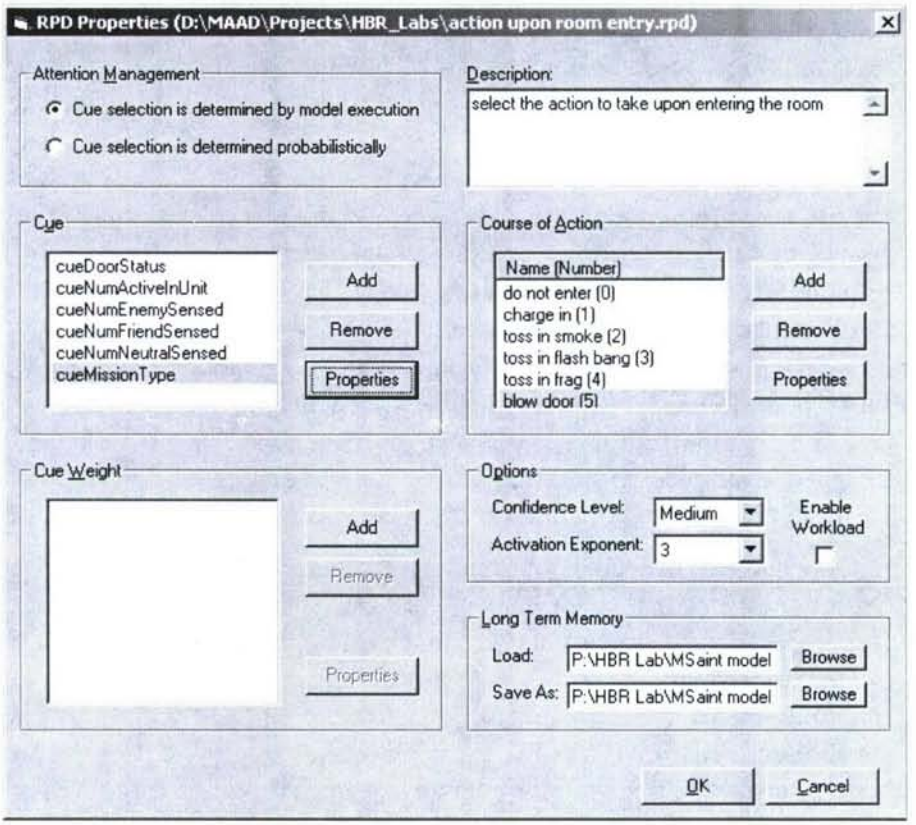

Figure 2-2. RPD tool properties interface.

Figure 2-2 shows a top-level interface used to specify the decision. The information supplied at this level specifies the gross structure of the decision, namely a list of the cues that prompt recognition, the process-level effects that impact recognition and the courses of action that follow recognition. By drilling down via the "Properties" button, the user can supply increasingly detailed information about the cues, processes and courses of action. For instance, the user must supply an "interpretation" of each cue either in terms of discrete ranges for real-valued cues or in terms of subjective labels for enumerated values. Similarly, the user has the option to describe how cue weights and discriminations change dynamically from situation to situation. Finally, the user must supply a set of conditions for each course of action that will be

used to determine when that course of action was successful. (At this point the user also has the option to describe the diagnostic reasoning associated with each course of action, and how that reasoning will feedback into the recognition process.) The remaining fields accept user inputs for various model parameters, secondary process effects (e.g., workload and attention management) and file management for long-term memory (including an option to run the model with a "pre-loaded" set of experiences).

2.2 Task Network Modeling (TNM)

One technology that has proven useful for predicting human-system performance is *task network modeling*. In a task network model, human performance of an individual performing a function (e.g., performing a procedure) is decomposed into a series of subfunctions, which are then decomposed into tasks. This is, in human factors engineering terms, the task analysis. The sequence of tasks is defined by constructing a *task network*. Complex task network models can include thousands of unique tasks with many interdependencies and complex decision logic.

Task analyses are organized by task sequence in the task network model. In addition to complex operator models, task network models can include sophisticated submodels of hardware and software to create a complete representation of the human/machine system (See Figure 2-3). Task network modeling is relatively easy to use and understand, and recent advancements in task network modeling technology have made it even more accessible to human factors practitioners. Finally, with a task network model, the human factors engineer can examine a design and address questions such as "How much longer will it take to perform this procedure?" and "Will there be a decrease in the error rate by using advanced technology equipment?"

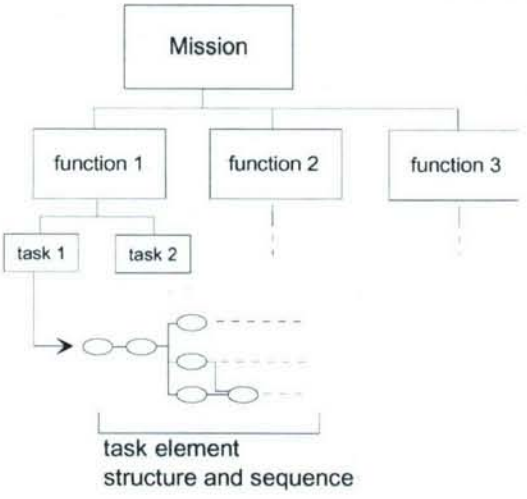

Figure 2-3. Concept of reductionist models of human performance.

Tools have emerged for modeling human performance in complex systems that evolve around the concept of task network modeling. Task network models of human performance begin with a functional decomposition of human activity (e.g., a task analysis). Then, by adding sequencing information, timing information and information on how human activity is related to other system behaviors, a model of human performance is created.

2.2.1 Micro Saint

Micro Saint is a commercial product that supports the development of task network models. Through a user-interface targeted at the human factors community, task network models of any size or complexity can be constructed.

Micro Saint is a tool that supports task network modeling and is ideal for modeling human behavior. Micro Saint will be used as the simulation engine for creating and analyzing models that represent human behavior. It is a discrete event simulation engine and runs under the Windows operating system.

2.2.1.1 Component Object Model (COM)

Component Object Model technology, a model for binary code developed by Microsoft, will be used as the protocol for communicating with Micro Saint. It permits variables within a Micro Saint model to be altered and observed by other applications.

With COM Services, a software application can receive variable and event queue data from Micro Saint along with messages indicating the status of model execution. Developing the structure for communication between Micro Saint and other software applications can be done in any programming environment that is COM-compliant, such as Visual C++ or Visual Basic.

COM Services includes the capability to:

- Pass variable values into and out of Micro Saint during model execution.
- Insert scenario events into the event queue at specified times in the future.
- Pause, halt, and abort the model through parsed expressions.
- Send messages to the user when a model has ended or if errors have occurred.

COM will be used as the means of communication between Micro Saint and the RPD tool, and between the Micro Saint model and the server middleware.

2.2.1.2 External Variables

Within the Micro Saint model, variables may be marked as being External Variables. A variable so marked will be transmitted via COM.

2.2.2 Interconnection with RPD

Micro Saint can also communicate with other applications through a fairly simple interface that is appropriate when the other application has no internal concept of time. This capability is referred to as an External Model Call (or EMC). This section describes techniques for implementing these connections.

Micro Saint communicates with external applications via COM. A unique derivate of COM is the EMC. When the COM capability was first initiated, the external application was required to be loaded and all communications were initiated via the external COM program. With EMCs, Micro Saint initiates the external communication as well as any further communication between Micro Saint and the external application. An EMC is defined in Micro Saint via a separate interface. With this interface, the user defines the external application name, the EMC name, and how Micro Saint will communicate to the external application. These communications act in three different ways; Micro Saint will only send variable information to the external application,

Micro Saint will only receive variable information from the external application, or Micro Saint will send variable information to the external application and expect the external application to send variable information back to Micro Saint. The variable information consists of the variable name along with its current value. Once the EMC is defined, it can be called anywhere in the model and as many times as the user needs. After an EMC is defined, when that Micro Saint model (which contains the EMC) is loaded into Micro Saint, Micro Saint will automatically load up the external application and initiate communication.

Typically, the external model has to be COM-enabled, and must be encapsulated in a way recognized by Micro Saint. To access the application, the External Model Call is embedded in the expression as though it were a user-defined macro or a variable name. When Micro Saint encounters the call during execution it pauses, starts the application, sends and receives the appropriate external variables, and either closes the external application or keeps it open depending on what is specified before the Micro Saint model continues. EMCs will be used by the Micro Saint model to control execution of the RPD tool.

2.3 Server: HBR Server

The server is a portion of the client-server architecture external to the SAF and is a stand-alone process that runs under the Windows operating system. Each server provides entity behavior attributes using DIS as a communications protocol. A server consists of two main components as shown in Figure 2-4; the middleware and a simulation engine. The simulation engine, Micro Saint, communicates with the middleware using Component Object Model (COM). Micro Saint contains models of the behaviors of interest and will have input and output variables marked as External Variables. These variables communicate with the DIS network and ultimately the SAF via the server middleware.

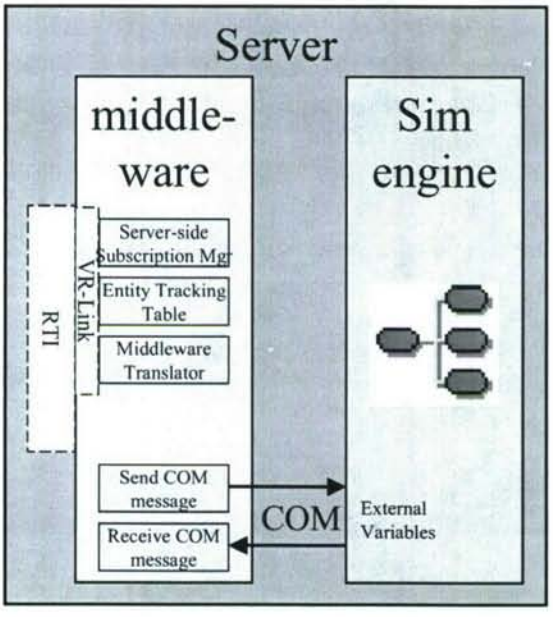

Figure 2-4. Server architecture.

2.3.1 Middleware

The middleware (Figure 2-5) acts as a data communications link between the DIS network and the simulation engine, Micro Saint. The middleware contains the following components: VR-Link, a subscription manager, a data handler, an entity tracking table, and a COM interface.

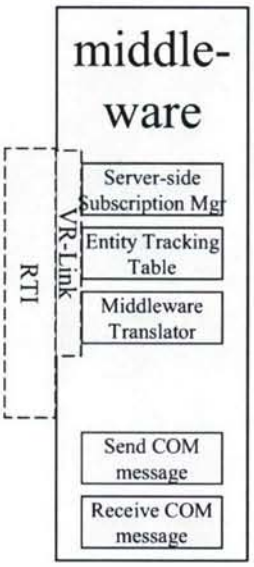

Figure 2-5. Middleware.

The Network Computer Interface (NCI) provides for DIS network communication between the server and the outside world. The NCI uses MäK Technologies' VR-Link Networking Toolkit to perform much of this functionality. The NCI is responsible for filtering out interactions that are not relevant to a specific server. The middleware is also responsible for the subscription of entities to a server, handling entity update requests, and sending entity task time and decision information. It also processes subscription requests for entities supported by this server. It uses VR-Link utilities to store entity information.

2.4 Client: SAF

The client portion of the client-server architecture consists of a SAF with three added processes (libraries) integrated in (Figure 2-6). The additional libraries are:

- Subscription manager
- Data handler
- Behavior

Figure 2-6. Client architecture.

2.4.1 Subscription process

The capability to subscribe an entity to a server has, on previous programs, been added to various SAFs. The subscription process occurs via a library named libsubscriptionmgr, the subscription manager library. This subscription process effectively transfers ownership of the SAF entity's behavior parameters to an external server.

The subscription process is a handshake protocol that provides a mechanism for load balancing, allows a client to locate a server that can provide the appropriate and desired service, and ensures that a one to one relationship is established between an entity requesting a service and the server providing that service. The libsubscriptionmgr library mechanizes the client side of the subscription process depicted in Figure 2-7. Functionally, once the subscription manager receives a request to subscribe an entity to a server, an Action Request is sent out asking, "Who can help me?" Anywhere from none to many servers may respond to that request with a corresponding Action Response indicating, "I can support you". The first response received will be chosen and then a second Action Request will be sent back to the responding server indicating, "I choose you." Only then will the final piece of the handshake protocol be sent as another Action Response from the server to the client indicating, "I acknowledge you choose me." This step completes the subscription process. Once this has occurred, the sending and receiving of data between the client and the server can occur.

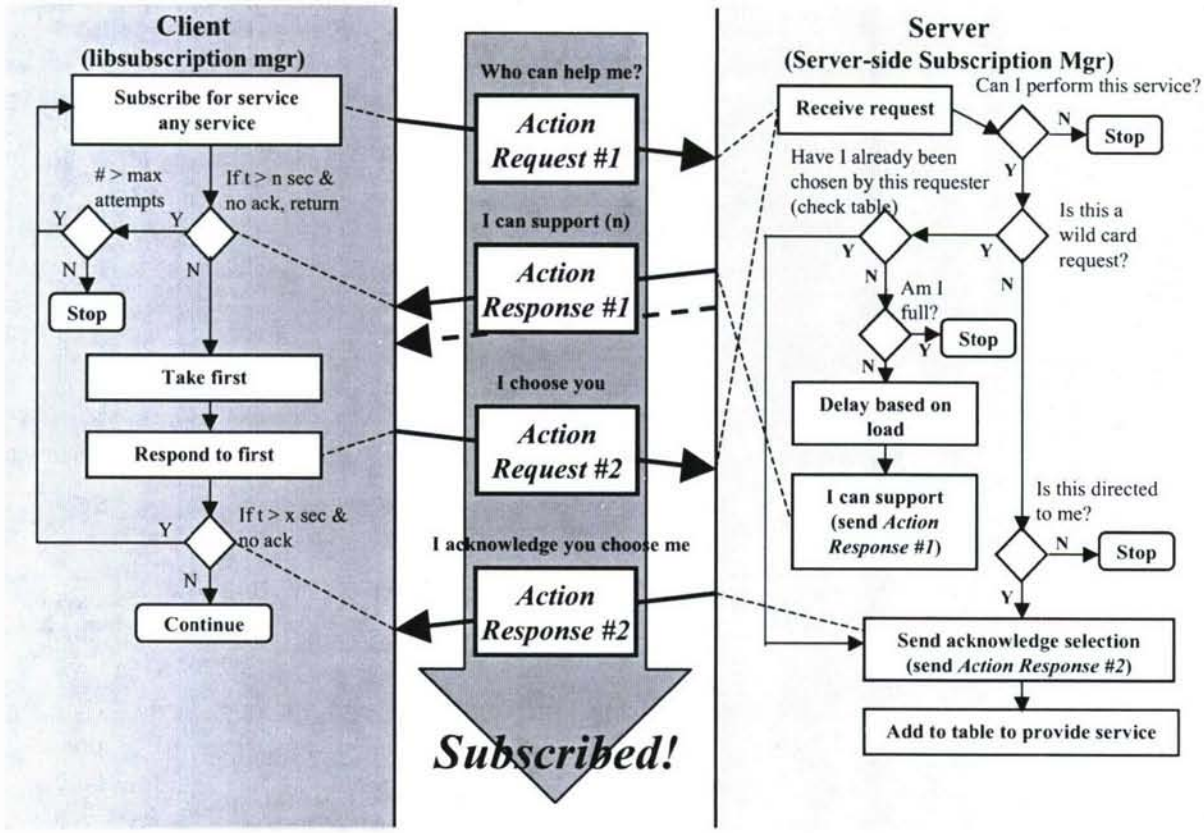

Figure 2-7. Client-Server subscription process.

2.4.2 Data Handling

Once an entity has subscribed to a specific server for a service, data must be transferred between the SAF and server. Within the SAF, the library that will perform this function is named libdatahandler (data handler library.) (See Figure 2-8.) This library will pack and unpack data packets passed between the client and the server. In the SAF, libraries containing the behaviors will supply the data values.

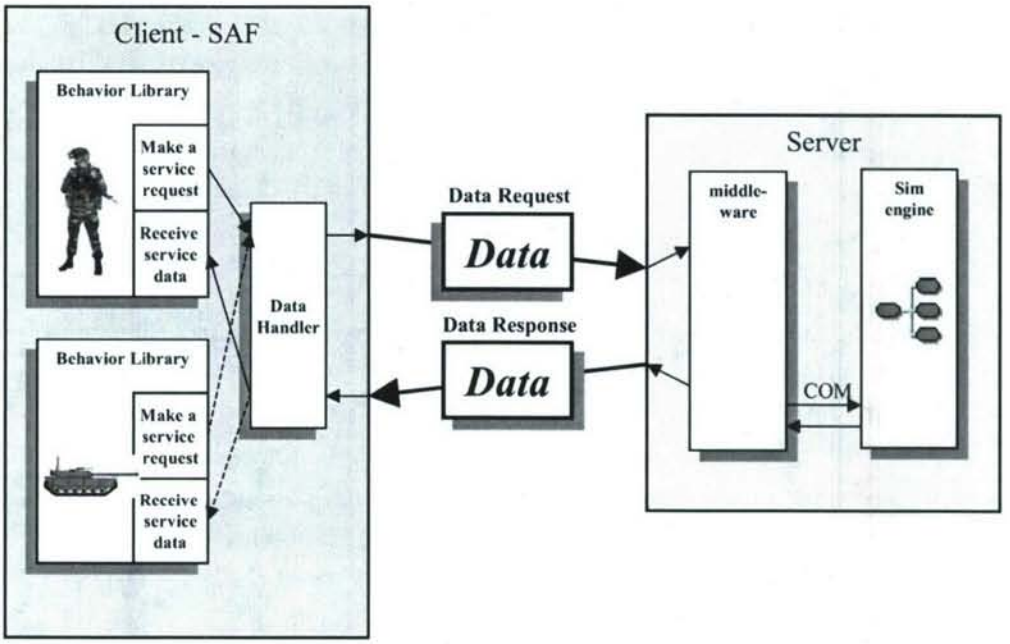

Figure 2-8. Client-Server data handling.

2.4.3 Behaviors

The third piece of the puzzle, which works in conjunction with libsubscriptionmgr and libdatahandler, is the libraries that contain the behaviors. These libraries contain the behaviors that will be affected by the high fidelity behavior models residing within the server. There are two basic types of behaviors that are currently being modified within a SAF: behaviors that relate to an amount of time to perform a task, and those surrounding/making decisions. These two types of behaviors, while both using the client-server architecture, can be implemented differently.

2.4.3.1 Time based behaviors

A SAF models the effects of some behaviors as tasks that require a specified amount of time to perform. The time to perform a task is often a function of various factors that describe the state of the environment in which the task is occurring. The time to perform a task is influenced by various factors and the time may represent many subtasks that actually occur. For example, the task of loading the main gun of a tank is modeled using a single task time. However, in reality, the load time includes multiple tasks that the tank crew members perform as well as the mechanical tasks carried out by the components of the tank. Time related behaviors can be determined in the server quickly with the desired task time returned to the SAF behavior before the task has completed, thus allowing the task time to be modified with the server derived time.

2.4.3.2 Decision based behaviors

An example of a decision based behavior is a tank commander choosing which firing position to change to: normal, hidden, or alternate. This type of decision is typically not associated with a task time but instead is a choice that must be made before continuing. (The time to make the decision is considered negligible and thus not modeled.)

2.4.4 Interconnection with Server

The method used by the client to contact the server involves a series of SIMAN (Simulation Management) interactions: "Action Request", "Action Response", and "Data". The client using these three SIMAN interactions initiates all data and subscription requests. This provides an efficient and clean request/response paradigm. The process is described in detail later in this report.

2.5 Client-Server Architecture

An architecture was needed that would support improving behavior representation of entities within Computer Generated Forces (CGFs). A client-server approach was identified as an alternative that would provide an architecture to implement behavioral performance without adversely affecting SAF performance.

The client-server architecture consists of a federation of disparate simulations. (See Figure 2-9.) These simulations are linked together to accomplish an overall objective of providing high fidelity behavior models that impact entity performance. A subscription process provides the capability for multiple SAFs to obtain behavior attributes from multiple behavior servers. This robust architecture was developed and extensively tested on previous programs. It should be noted that while the underlying architecture is used here to incorporate improved behaviors into CGFs, the same architecture and technologies could be used to incorporate other effects that could impact a CGF entities' ability to perform actions.

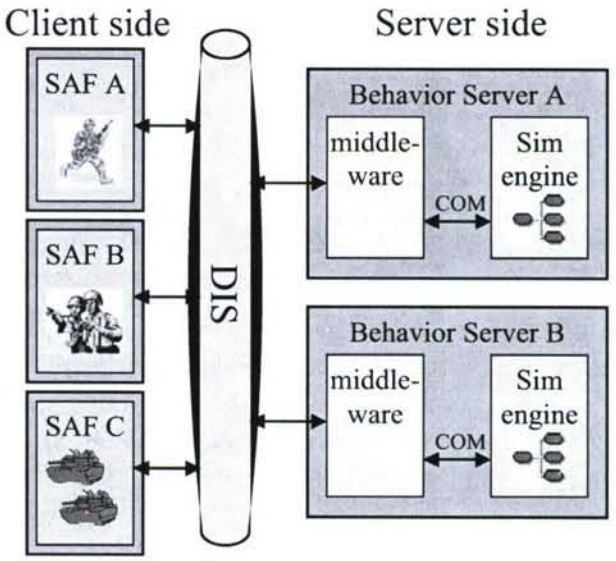

Figure 2-9. Client-Server federation.

For improving behavioral representation into a SAF or other CGF application the client-server approach is a flexible architecture to incorporate behavioral representation or other functionality. A major benefit is that once the server interface is incorporated, the server can be modified without affecting the CGF application.

We incorporated entity behavior models into a "behavior server" and connected it with a "client" which is the entity-based simulation (for this program the SAF). The "server" tracks simulation entities and provides the client with operational parameters for the simulation to use for entities

that it is representing. This approach allows for highly complex behaviors to be modeled externally in a server and represented in an entity-based simulation.

The architecture of the behavior server and SAFs is depicted in Figure 2-10. The client SAFs have been fitted with the three new features: a subscription processor, a data handing capability, and a new server ready behavior library. Each behavior server contains a processing engine (Micro Saint has been used but the architecture does not preclude others) and a Network Computer Interface (NCI) to communicate using DIS protocols. This architecture is scalable and allows for any number of servers to be present.

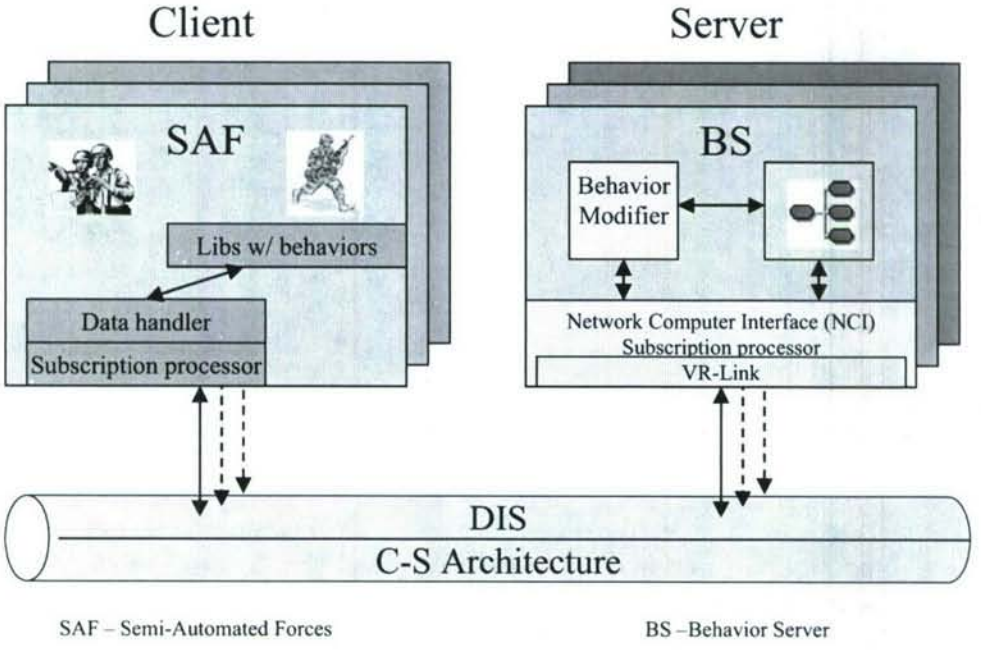

Figure 2-10. Client-Server architecture.

Entity subscription and data handling have been integrated into the chosen CGFs (i.e., DISAF and OTB-JVB).

2.5.1 Mapper

A tool has been created to facilitate the mapping of parameter between the server middleware and SAF. Each parameter that is communicated must have a one-to-one mapping between the two. A screenshot of the user interface of the Mapper that contains an example mapping between DISAF variables and Micro Saint variables for the building clearing model is shown in Figure 2-11.

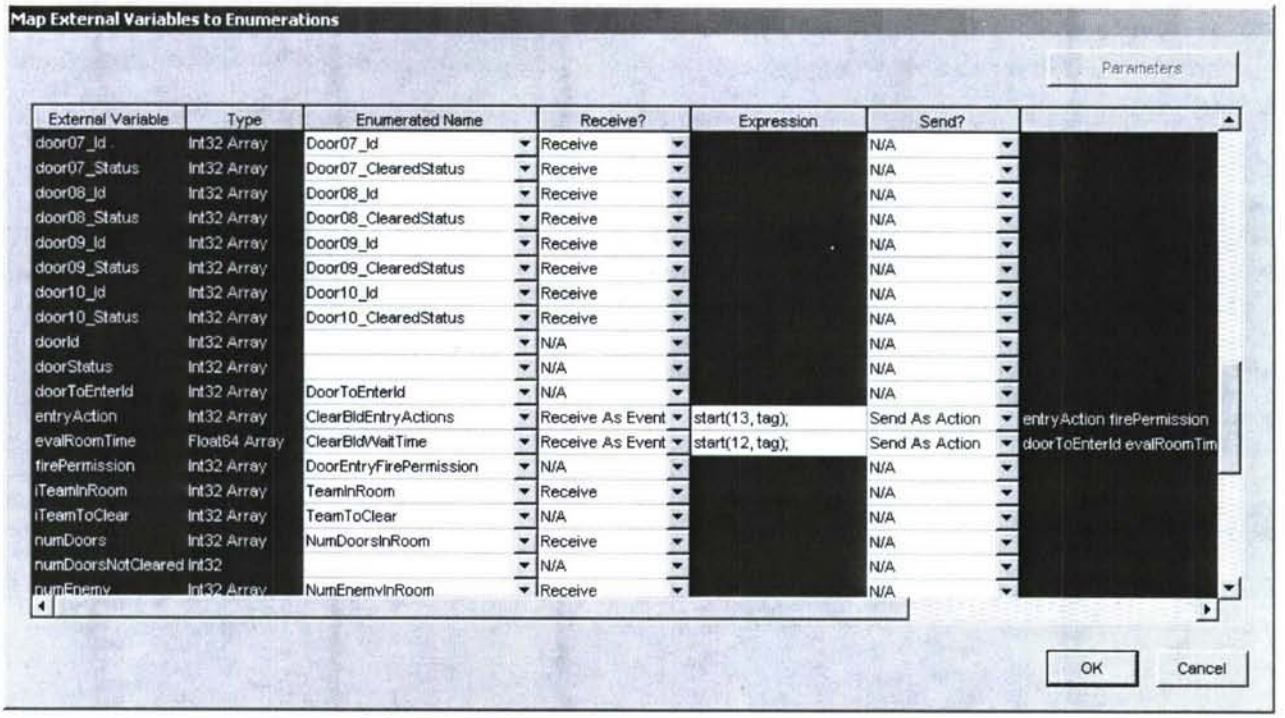

Figure 2-11. Mapper user interface.

3.0 Modeling Approach

The focus of this effort was to provide realistic behaviors for dismounted infantry entities as they clear a building of possible hostiles and neutrals. The model should allow the soldiers to navigate through the building, engage hostiles as necessary, and secure the building, all using realistic human behaviors. The soldiers were also given the capability to use "smart sensor web" type devices to allow them to gain additional situation awareness about the contents of the rooms to be cleared. Sensors modeled included a daylight TV and a forward looking infrared (FLIR) camera.

3.1 RPD in the Clear Building model

A unit of dismounted infantry personnel performs the operations necessary to clear a building. As they encounter numerous situations, either individually or collectively, they are required to make decisions. RPD was used to model the cognitive making process involved with selected decisions encountered while the squad clears the building.

For each decision that the RPD is called upon to process, relevant situational awareness (SA) data is required. The SA data acts as cues upon which the decision will be based. RPD obtains all SA data via COM messages from the Micro Saint model.

For the clear building operation, a few key decisions were modeled using RPD. The RPD model for clearing a building is named "Clear Bld using RPD" and it contains three decisions:

- door selection
- action to take upon room entry, and
- setting of fire permission.

3.1.1 Door selection

As a fireteam moves from room to room to clear the entire building, decisions are made as to which door to proceed through next. The decision that determines which door in the current room to enter next is made for each fireteam using the RPD decision named, "evaluate this door". The fireteam is presented with one door at a time and it determines, based upon SA data, to select or not select that door as the next door to enter.

The SA data used for the door evaluation decision are the following

- Door status: cleared or not cleared (Table 3-1)
- Number of doors in the room (Table 3-2)

Table 3-1. Encoding of door cleared status.

Enumeration	Alias	Meaning
0	not cleared	Door not cleared
1	1st door cleared	1st door cleared
2	2nd door cleared	2nd door cleared
i	ith door cleared	ith door cleared
500000	500000th door cleared	500000 (max) door cleared

Table 3-2. Encoding of the number of doors in the room.

Enumeration	Alias	Meaning
0 <= value < 2	one	Room with 1 door
2 <= value < 3	two	Room with 2 doors
3 <= value <= 100	some	Room with 3 or more doors

Included with this decision are traces in LTM. Table 3-3 indicates the traces that were placed into LTM and are present prior to the first encounter with this decision. As each encounter with this decision occurs, the trace of it is added to LTM and will be used in successive decisions of this type.

Table 3-3. LTM traces for door selection.

# of traces	Cue door status	Cue # doors	COA	Success
10	Cleared	One	Select	Yes
10	Cleared	One	Do not select	No

# of traces	Cue door status	Cue # doors	COA	Success
10	Not cleared	One	Select	Yes
10	Not cleared	One	Do not select	No
5	Not cleared	Two	Select	Yes
5	Cleared	Two	Select	No
2	Cleared	Two	Select	Yes
5	Not cleared	Some	Select	Yes
10	Cleared	Some	Do not select	Yes
2	Cleared	Some	Select	Yes

3.1.2 Action upon room entry

Once a fireteam has selected the door to enter next, a decision is made as to the method of entry that should be used. The decision that determines the action to take upon entering the next room is made for each fireteam using the RPD decision named, "action upon room entry". The fireteam is presented with the door, any known information about what type of people are in the room, and the type of mission being performed, and it determines, based upon SA data, the method of entry.

The decision of "action upon room entry" selects the action to take upon entering the room. The resulting decision is either to not enter at all and abort the mission, to charge in, or to first throw in some type of device and then enter the room.

The SA data used for the door evaluation decision are the following

- Door status: cleared or not cleared (Table 3-4)
- Number of members still active in this unit (Table 3-5)
- Number of enemy sensed beyond the selected door (Table 3-6)
- Number of friendly sensed beyond the selected door (Table 3-7)
- Number of neutral sensed beyond the selected door (Table 3-8)
- Mission type (Table 3-9)

Table 3-4. Encoding of door cleared status.

Enumeration	Alias	Meaning
0	not cleared	Door not cleared
1	1st door cleared	1st door cleared
2	2nd door cleared	2nd door cleared

Enumeration	Alias	Meaning
I	ith door cleared	ith door cleared
500000	500000^{th} door cleared	500000 (max) door cleared

Table 3-5. Encoding of the number of active members in the unit

Enumeration	Alias	Meaning
1 <= value < 3	Couple	At least two members of the unit are sill active
3 <= value < 100	More than 2	More than two members of the unit are still active

Table 3-6. Encoding of the number of enemy sensed in the next room.

Enumeration	Alias	Meaning
0 <= value < 1	None	No enemy sensed
1 <= value < 3	Some	Some, one or two, enemy sensed
3 <= value <= 10000	Lots	Lots of, three or more, enemy sensed

Table 3-7. Encoding of the number of friendly sensed in the next room.

Enumeration	Alias	Meaning
0 <= value < 1	None	No friendlies sensed
1 <= value < 3	Some	Some, one or two, friendly sensed
3 <= value <= 10000	Lots	Lots of, three or more, friendly sensed

Table 3-8. Encoding of the number of neutrals sensed in the next room.

Enumeration	Alias	Meaning
0 <= value < 1	None	No neutrals sensed
1 <= value < 3	Some	Some, one or two, neutrals sensed
3 <= value <= 10000	Lots	Lots of, three or more, neutrals sensed

Table 3-9. Encoding of the mission type.

Enumeration	Alias	Meaning
0	Rescue	Rescue hostages
1	Seize	Seize the building

Included with this decision are traces in LTM. Table 3-10 indicates the traces that were placed into LTM and are present prior to the first encounter with this decision. As each encounter with this decision occurs, the trace of it is added to LTM and will be used in successive decisions of this type.

Table 3-10. LTM traces for action upon room entry.

# of traces	Cue door status	Cue # active in unit	Cue # enemy sensed	Cue # friend sensed	Cue # neutral sensed	Cue mission type	COA	Success
10	Not cleared	Couple	Lots	None	Some	Rescue	Do not enter	Yes
5	Not cleared	Couple	Lots	None	Some	Seize	Do not enter	Yes
5	Not cleared	More than 2	Lots	None	Some	Seize	Charge in	No
10	Cleared	Couple	None	None	None	Seize	Charge in	Yes
10	Cleared	Couple	None	None	None	Seize	Do not enter	No
10	Not cleared	More than 2	None	None	None	Seize	Charge in	Yes
5	Not cleared	More than 2	Some	Some	Some	Seize	Do not enter	No
5	Not cleared	More than 2	Lots	Some	Some	Seize	Do not enter	No
10	Not cleared	More than 2	Some	None	None	Seize	Toss Frag	Yes
10	Not cleared	More than 2	Some	None	None	Rescue	Blow door	Yes

3.1.3 Fire permission

Once a fireteam has selected the door to enter next, a decision is made to set the fire permission. The decision that determines the fire permission upon entering the next room is made for each fireteam using the RPD decision named, "fire permission". The fireteam is presented with the door, any known information about what type of people are in the room, and the method of entry, and it determines, based upon SA data, the fire permission for the unit.

The decision of "fire permission" sets the fire permission for the chosen room entry action. The resulting decision is to set the fire permission to either hold, tight, or free.

The SA data used for the fire permission decision are the following

- Entry action (Table 3-11)
- Door status: cleared or not cleared (Table 3-12)
- Number of members still active in this unit (Table 3-13)
- Number of enemy sensed beyond the selected door (Table 3-14)
- Number of friendly sensed beyond the selected door (Table 3-15)
- Number of neutral sensed beyond the selected door (Table 3-16)

Table 3-11. Encoding of the entry action.

Enumeration	Alias	Meaning
0	Do not enter	Do not enter the room, abort the mission
1	Charge in	Charge in
2	Toss smoke	Toss in smoke and then charge in
3	Toss flash	Toss in flash bang and then charge in
4	Toss frag	Toss in fragmentary grenade and then charge in
5	Blow door	Blow door and then charge in

Table 3-12. Encoding of door cleared status.

Enumeration	Alias	Meaning
0	not cleared	Door not cleared
1	1st door cleared	1st door cleared
2	2nd door cleared	2nd door cleared
I	ith door cleared	ith door cleared
500000	500000th door cleared	500000 (max) door cleared

Table 3-13. Encoding of number of active members in this unit.

Enumeration	Alias	Meaning
0 <= value < 1	None	No enemy sensed
1 <= value < 3	Some	Some, one or two, enemy sensed
3 <= value <= 10000	Lots	Lots of, three or more, enemy sensed

Table 3-14. Encoding of the number of enemy sensed in the next room.

Enumeration	Alias	Meaning
0 <= value < 1	None	No enemy sensed
1 <= value < 3	Some	Some, one or two, enemy sensed
3 <= value <= 10000	Lots	Lots of, three or more, enemy sensed

Table 3-15. Encoding of the number of friendly sensed in the next room.

Enumeration	Alias	Meaning
0 <= value < 1	None	No friendlies sensed
1 <= value < 3	Some	Some, one or two, friendly sensed
3 <= value <= 10000	Lots	Lots of, three or more, friendly sensed

Table 3-16. Encoding of the number of neutrals sensed in the next room.

Enumeration	Alias	Meaning
0 <= value < 1	None	No neutrals sensed
1 <= value < 3	Some	Some, one or two, neutrals sensed
3 <= value <= 10000	Lots	Lots of, three or more, neutrals sensed

Included with this decision are traces in LTM. Table 3-17 indicates the traces that were placed into LTM and are present prior to the first encounter with this decision. As each encounter with this decision occurs, the trace of it is added to LTM and will be used in successive decisions of this type.

Table 3-17. LTM traces for fire permission.

# of traces	Cue entry action	Cue door status	Cue # active in unit	Cue # enemy sensed	Cue # friend sensed	Cue # neutral sensed	COA	Success
10	Charge in	Cleared	Some	None	None	Lots	FP tight	Yes
10	Charge in	Cleared	Some	Some	None	Some	FP hold	No
10	Charge in	Cleared	Some	Lots	None	Some	FP hold	No

# of traces	Cue entry action	Cue door status	Cue # active in unit	Cue # enemy sensed	Cue # friend sensed	Cue # neutral sensed	COA	Success
10	Charge in	Not cleared	Some	None	None	None	FP free	Yes
10	Charge in	Not cleared	Some	Some	None	Some	FP free	Yes

3.1.4 Interconnection with Micro Saint model

Based upon these SA data, and previous experience in making this type of decision (as stored in long term memory), the resulting Course of Action (COA) is determined. Each resulting COA is sent back to the Micro Saint model in response to the request for the decision.

For the decision of door selection, two COAs were possible:

- do not select this door
- select this door

Table 3-18 shows the set of rules used as the criteria to determine if the door selection COA selected was a successful outcome.

Table 3-18. Success criteria for door selection.

COA	COA Enum	Success Criteria
Do not select this door	0	(rpdNumDoors <> 1) & (rpdDoorStatus <> DOOR_NOT_CLEARED) & (rpdNumDoors <> rpdNumDoorsCleared | rpdDoorStatus = DOOR_NOT_CLEARED)
Select this door	1	(rpdNumDoors = 1) | (rpdDoorStatus = DOOR_NOT_CLEARED) | (rpdNumDoors = rpdNumDoorsCleared)

For the decision of entry method, six COAs were possible:

- do not enter
- charge in
- toss in smoke and then charge in
- toss in flash bang and then charge in
- toss in fragmentary grenade and then charge in
- blow door and then charge in

Table 3-19 shows the set of rules used as the criteria to determine if the entry method COA selected was a successful outcome.

Table 3-19. Success criteria for room entry method.

COA	COA Enum	Success Criteria
Do not enter	0	rpdDoorStatus = DOOR_NOT_CLEARED & rpdNumInThisUnit < rpdNumEnemy
Charge in	1	rpdDoorStatus <> DOOR_NOT_CLEARED | rpdNumEnemy = 0 | rpdNumFriend >= 1
Toss in smoke	2	(rpdDoorStatus <> DOOR_NOT_CLEARED & rpdNumNeutral >= 1 & rpdNumEnemy >= 1) | (rpdDoorStatus <> DOOR_NOT_CLEARED & rpdNumNeutral >= 1)
Toss in flash bang	3	rpdDoorStatus = DOOR_NOT_CLEARED & rpdNumNeutral >= 1 & rpdNumEnemy >= 1
Toss in frag	4	rpdDoorStatus = DOOR_NOT_CLEARED & rpdNumFriend = 0 & rpdNumNeutral = 0 & rpdNumEnemy >= 1
Blow door	5	rpdDoorStatus = DOOR_NOT_CLEARED & rpdNumFriend = 0 & rpdNumNeutral <= 1

For the decision of fire permission, three COAs were possible:

- Fire permission hold
- Fire permission tight
- Fire permission free

Table 3-20 shows the set of rules used as the criteria to determine if the entry method COA selected was a successful outcome.

Table 3-20. Success criteria for fire permission.

COA	COA Enum	Success Criteria
FP hold	0	rpdDoorStatus <> DOOR_NOT_CLEARED | (rpdNumNeutral >= 1 & rpdNumEnemy = 0) | rpdNumFriend >= 1
FP tight	1	cueEntryAction = 0 | (rpdNumEnemy < rpdNumInThisUnit & rpdNumNeutral >= 1)
FP free	2	rpdDoorStatus = DOOR_NOT_CLEARED | ((rpdNumEnemy >= 1 & rpdNumFriend = 0 & rpdNumNeutral <= 1) | cueEntryAction > 1)

3.2 Task Network Modeling

Task network models will be built to simulate behavioral characteristics of selected tasks. These task network models will reside in Micro Saint models in the server and will provide behavior

attributes to SAF entities on an as needed basis. Development of these models is dependent on the specific behaviors being composed.

3.2.1 Micro Saint model – Clear Building

Micro Saint was developed and has been used specifically for human behavior representation and was used here to model tasks and decisions a squad of dismount infantry would encounter when clearing a building. Micro Saint will reside in the server and will perform analysis of behaviors models as they are requested by the middleware. The simulation engine will act as a transfer function by feeding SAF supplied parameters into a model of a behavior and calculating desired behavior parameters as shown in Figure 3-1. A service request originating in the SAF indicates the desired behavior model and decisions that are needed.

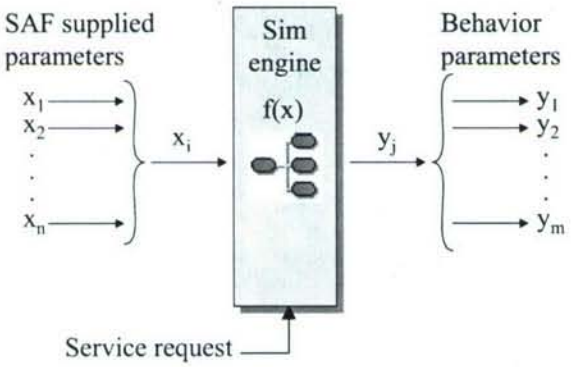

Figure 3-1. Micro Saint as a simulation engine.

The complex cognitive processing involved with clearing a building can be more effectively modeled using a task networking simulation tool such as Micro Saint, rather than within the confines of an entity simulation such as a SAF.

The Micro Saint model for clearing a building consists of fireteam tasks and decisions that would be taken to clear a building. The model is divided into responding to two service requests, which are:

- Service request #12 - Time to evaluate the room, door selection, and fireteam selection
- Service request #13 - Action to take upon entering the room, and the fire permission

Figure 3-2 is the top-level task network diagram of the Micro Saint model for a DIs clearing a building. This model handles the two services requests and provides four decisions and one task time.

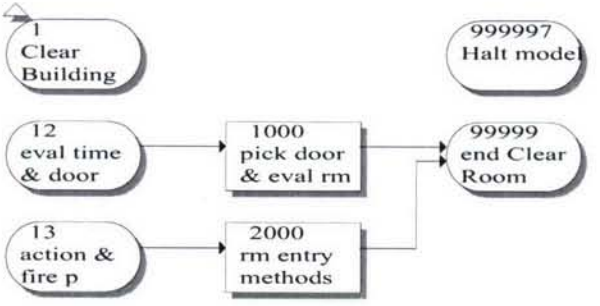

Figure 3-2. Top-level network of "Clear Bld using RPD".

The TNMs for the two service requests are depicted as rectangular boxes and are labeled, "pick door & eval rm", and "rm entry methods". A discussion of each of these sub-networks is given below.

Network 1000, "pick door & eval rm", selects the door to enter next, the fireteam to enter it, and the time needed to evaluate the room at the stack point. This network contains two sub-networks and a data collection task and is shown in Figure 3-3.

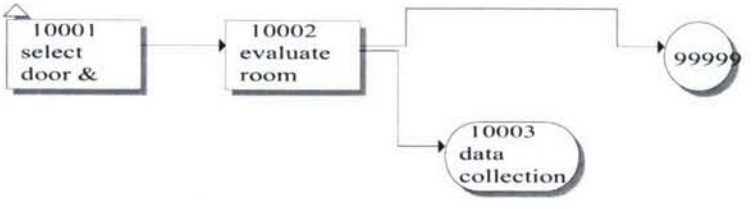

Figure 3-3. TNM of door selection and time to evaluate room.

The sub-network to select a door and the fireteam is shown in Figure 3-4. It selects the entry point (door) and the fire team to breach it, and then once at the stack point has them perform the task of interrogating the room.

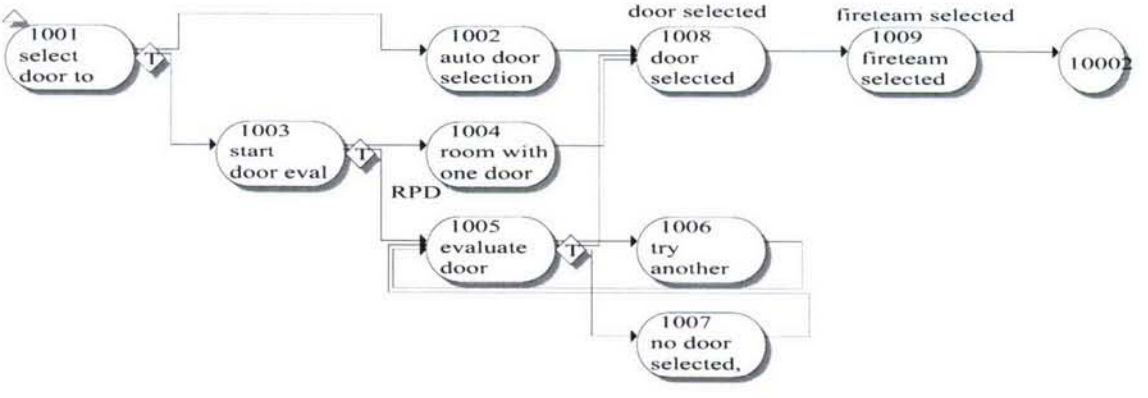

Figure 3-4. Sub-network of door and fireteam selection.

The sub-network to perform the room evaluation task is shown in Figure 3-5.

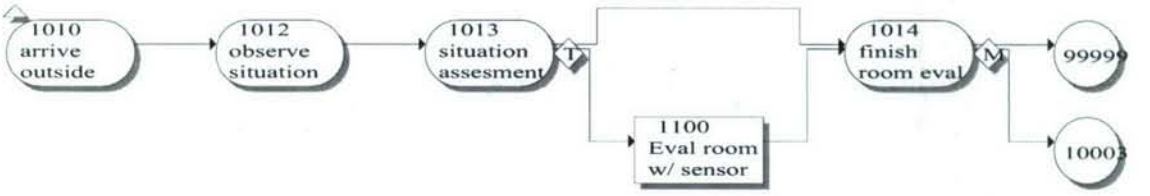

Figure 3-5. Sub-network of room evaluation.

The sub-network to evaluate the room using a sensor is shown in Figure 3-6. The model can be configured to use one of the smart-sensor web devices or to not use a sensor at all.

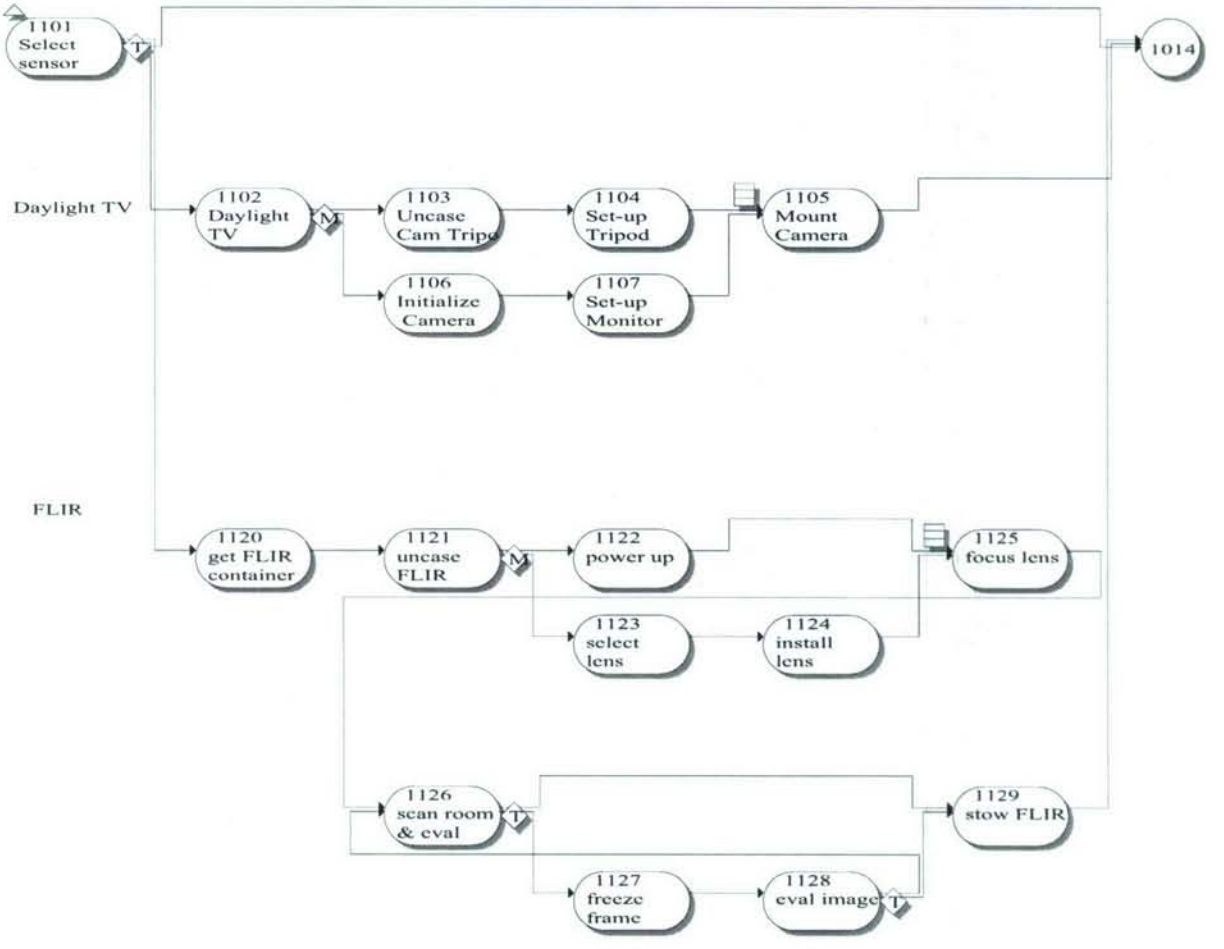

Figure 3-6. Sub-network of room evaluation with sensor.

The sub-network to perform the selection room entry method is shown in Figure 3-7. This sub-network is used to figure out the method of entering the room and the action to take, including the fire permission based on SA gathered.

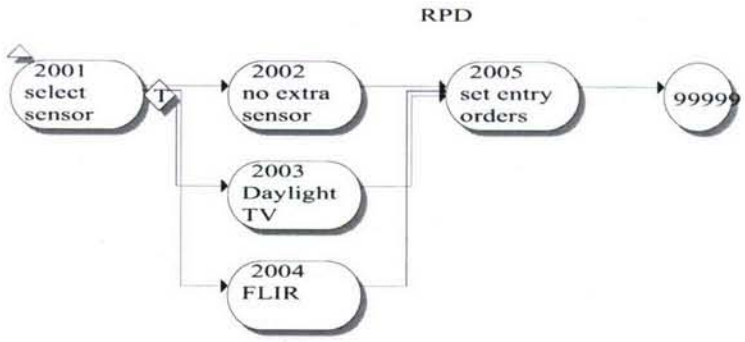

Figure 3-7. Sub-network of room entry methods.

The clear building model has a set of variables that are currently manually set in the Micro Saint model but in the future they could be obtained from the SAF. At this time DISAF and OTB-JVB do not contain these parameters. They are listed in Table 3-21 below:

Table 3-21. Clear building options.

Variable	Description	Default	Enumerations
decisionApp	Decision application tool. Determines the application used to determine the decisions.	RPD	MICRO_SAINT RPD
missionType	Clear building mission type.	MISSION_SEIZE	MISSION_RESCUE MISSION_SEIZE
sensorSelected	Sensor to use to gain situational awareness information about the contents of the room.	SENSOR_DTV	SENSOR_NONE SENSOR_DTV SENSOR_FLIR
senseNeeded	Should a sensor be used?	TRUE	FALSE TRUE

The sensors have detection characteristics that characterize their ability to gather situational awareness information about the inhabitants of the room being sensed. The characteristics are shown in Table 3-22 (example data is shown, actual values may be different) and are used to determine the number of friendly, enemy, and neutral individuals are sensed in the room based on truth data.

Table 3-22. Sensor detection characteristics.

Sensor	Probability of detection, Pd	Probability of false alarm, Pfa	Probability of correct classification, Pcc
None	0.1	0.25	0.2

Sensor	Probability of detection, Pd	Probability of false alarm, Pfa	Probability of correct classification, Pcc
Daylight TV	0.96	0.04	0.88
FLIR	0.94	0.04	0.86

3.2.2 I/O with SAF

For each type of service request available to the client SAF, a corresponding service response must exist in the server model. The correlation of those two items is accomplished via a mapping table. Table 3-23 shows the two service request types that will be handled by the clear building model which reside in Micro Saint. One is the request to evaluate the room at the stack point and corresponds to service request #12. In response to this service request the task network model for clearing a building will be activated. After processing the situation awareness data, three variables indicating the ID of the door to enter next, the time to spend at the stack point evaluating the room, and the index of the fireteam to enter the room are determined and submitted back.

Table 3-23. Middleware to Micro Saint model mapped events.

External Variable	Type	Enumerated Name	Expression	Variables
evalRoomTime	Float 64 array	ClearBldWaitTime	start(12, tag);	doorToEnterId evalRoomTime iTeamToClear
entryAction	Int 32 array	ClearBldEntryActions	start(13, tag);	entryAction firePermission

Similarly, for service request #13, the action to take upon entering the room is requested and two variables describing the type of entry action to take and the fire permission setting are determined and submitted back.

Table 3-24 and Table 3-25 show the I/O for service request 12, Clear Room Eval Time.

Table 3-24. SAF to Middleware parameters for Service Request 12.

SA data from SAF	Comment
numDoors	Number of doors in the room
doorID[0-9]	ID numbers of each door (max of 10 doors per room)
doorStatus[0-9]	Status of each door (max of 10 doors per room)
numFireteams	Number of fireteams still active
iTeamInRoom	Index of fireteam in the room

Table 3-25. Middleware to SAF parameters for Service Request 12.

Responses to SAF	Comment
evalRoomTime	Time to evaluate the room
doorToEnterId	ID of the door to enter next
iTeamToClear	Index of fireteam selected to clear the next room

Table 3-26 and Table 3-27 show the I/O for service request 13, Clear Building Entry Actions.

Table 3-26. SAF to Middleware parameters for Service Request 13.

SA data from SAF	Comment
doorStatus[0-9]	Status of each door (max of 10 doors per room)
ammountAmmo	Amount of ammunition for fireteam
numEnemy	Number of enemy in the room
numNeutral	Number of neutrals in the room
numFriend	Number of friendly in the room
numInThisUnit	Number of active members of this fireteam

Table 3-27. Middleware to SAF parameters for Service Request 13.

Responses to SAF	Comment
firePermission	Fire permission setting
entryAction	Action to take to enter the room

A mapping of parameters and service request/responses exists to coordinate passing of information between the client and the server. Figure 3-8 shows the flow of information between DISAF and the clear building model residing in the server. This transfer of data is coordinated by the middleware. DISAF initiates the flow of data by making a service request that would typically include associated situation awareness parameters. These parameters are sent to the middleware that decodes the service request and passes the SA data via COM along to the behavior server model of clearing a building. The Micro Saint model is then prompted with the appropriate service request, executes the clear building model, and responds by returning the appropriate decisions back to DISAF, again through the middleware.

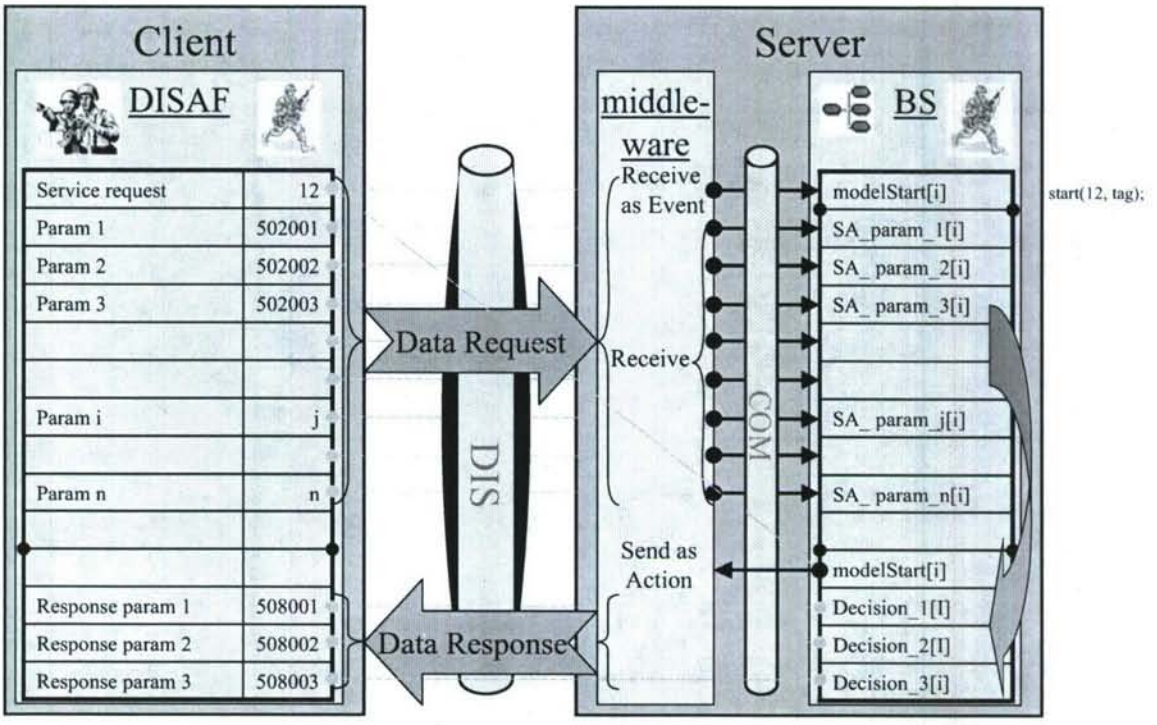

Figure 3-8. Clear Building I/O Map.

Table 3-28 contains a list of the mapping of variables used in the clear building model. Situational awareness parameters and data determined in response to a service request must be mapped between those used in the Micro Saint model and those coming from the client that passes through the middleware. The table indicates the cross referencing of those variables. The variable names in the column labeled "External Variable" are the names of the variables residing in the Micro Saint model of clearing a building. The corresponding variable names in the column labeled "Enumerated Name" are the names of the variables as mapped in the server middleware. Each of these variables matches a variable being communicated via DIS (or HLA) from the middleware to the client SAF.

Table 3-28. Clear building model to middleware I/O.

External Variable	Type	Enumerated Name
door01_Id	Int 32 array	Door01_Id
door02_Id	Int 32 array	Door02_Id
door03_Id	Int 32 array	Door03_Id
door04_Id	Int 32 array	Door04_Id
door05_Id	Int 32 array	Door05_Id
door06_Id	Int 32 array	Door06_Id
door07_Id	Int 32 array	Door07_Id
door08_Id	Int 32 array	Door08_Id

External Variable	Type	Enumerated Name
door09_Id	Int 32 array	Door09_Id
door10_Id	Int 32 array	Door10_Id
door01_Status	Int 32 array	Door01_ClearedStatus
door02_Status	Int 32 array	Door02_ClearedStatus
door03_Status	Int 32 array	Door03_ClearedStatus
door04_Status	Int 32 array	Door04_ClearedStatus
door05_Status	Int 32 array	Door05_ClearedStatus
door06_Status	Int 32 array	Door06_ClearedStatus
door07_Status	Int 32 array	Door07_ClearedStatus
door08_Status	Int 32 array	Door08_ClearedStatus
door09_Status	Int 32 array	Door09_ClearedStatus
door10_Status	Int 32 array	Door010_ClearedStatus
doorToEnterId	Int 32 array	DoorToEnterId
entryAction	Int 32 array	ClearBldEntryActions
evalRoomTime	Float 64 array	ClearBldWaitTime
firePermission	Int 32 array	DoorEntryFirePermission
iTeamInRoom	Int 32 array	TeamInRoom
iTeamToClear	Int 32 array	TeamToClear
numDoors	Int 32 array	NumDoorsInRoom
numEnemy	Int 32 array	NumEnemyInRoom
numFireteams	Int 32 array	NumFireteams
numFriend	Int 32 array	NumFriendlyInRoom
numInThisUnit	Int 32 array	NumInMyUnit
numNeitral	Int 32 array	NumNeutralInRoom

3.2.3 I/O with RPD

This section contains the Micro Saint to RPD I/O for the clear building model for each decision.

1. evaluate this door
    a. cueDoorStatus
    b. cueNumDoors
    c. DOOR_NOT_CLEARED

d. DOOR_CLEARED
e. rpdDoorStatus
f. rpdNumDoors
g. rpdNumDoorsCleared

2. action upon room entry
    a. cueAmountAmmo
    b. cueDoorStaus
    c. cueMissionType
    d. cueNumEnemySensed
    e. cueNumFriendSensed
    f. cueNumInThisUnit
    g. cueNumNeutralSensed
    h. DOOR_JUST_CLEARED
    i. DOOR_NOT_CLEARED
    j. MISSION_SEIZE
    k. rpdAmountAmmo
    l. rpdDoorStatus
    m. rpdNumEnemy
    n. rpdNumFriend
    o. rpdNumInThisUnit
    p. rpdNumNeutral
3. set fire permission
    a. cueDoorStatus
    b. cueEntryAction
    c. cunNumActiveInUnit
    d. cueNumEnemySensed
    e. cueNumFriendSensed
    f. cueNumNuetralSensed
    g. DOOR_JUST_CLEARED
    h. rpdDoorStatus
    i. rpdNumEnemy
    j. rpdNumFriend

k. rpdNumInThisUnit

l. rpdNumNeutral

3.2.4 EMCs with RPD

EMCs define the connection of the Micro Saint model of clearing a building to the RPD tool and its model of decisions used in clearing the building. Four EMCs were used to interface and control RPD and they are:

- EMCStaRPD – starts RPD
- EMCDecRPD – indicates the selected decision from RPD
- EMCExpRPD – sends expectancies to RPD
- EMCRetRPD – returns a Course of Action (COA) for the selected decision from RPD to the Micro Saint model

When a decision is requested by using the EMCDecRPD function, an ordered list of variables is sent via COM from the Micro Saint model to the RPD model. For the clear building model the variables in Table 3-29 were sent.

Table 3-29. RPD decision request variables.

Order	Variable Name
1	RPDClock
2	cueDoorStatus
3	cueNumDoors
4	rpdNumDoors
5	rpdDoorStatus
6	DOOR_CLEARED
7	MISSION_SEIZE
8	rpdNumDoorsCleared
9	cueNumActiveInUnit
10	cueNumEnemySensed
11	cueNumFriendSensed
12	cueNumNeutralSensed
13	cueMissionType
14	rpdNumInThisUnit
15	rpdNumEnemy
16	rpdNumFriend
17	rpdNumNeutral

Order	Variable Name
18	cueEntryAction
19	DOOR_NOT_CLEARED
20	RPDDecEMC
21	cueEarlyDoor

3.3 Server – HBR Server (w/Micro Saint HPM, RPD, and Middleware)

The HBR Server consists of the following:

- Micro Saint VR-Link SAF DIS Middleware for HBR Lab
- Simulation engine
- RPD

3.4 Client – DISAF or OTB-JVB

For this effort, the clients are DISAF v7.1 and the OTB-JVB baselines. Both SAFs operate in a DIS environment and can interact with the same "Clear Building" behavior model.

3.4.1 Added behaviors for clear building

While numerous SAF enhancements were required to support the "Clear Building" behavior, only one new library was needed: libusqclrbldg. This library is responsible for all the core activities involved with an infantry squad clearing a building.

3.4.2 Create a SAF behavior for improved behavioral representation

The third process, behavior attribute modification, requires additions and changes to behavior algorithms to allow for the use of server supplied behaviors in place of the standard semi-automated force (SAF) supplied attribute values. In this case, a new behavior was developed from scratch with the server-supported capabilities built in.

One of the major technical challenges of this program was developing a baseline SAF behavior capable of working with a server. Previously within DISAF, the "Automated Clear Room" behavior was altered to allow for server provided instruction. In the course of this work, we noticed a number of decisions that the SAF was forcing onto a user, or that were just not being factored into the behavior. Given the lack or realism and control this provided, the decision was made to create an infantry squad level "Clear Building" behavior. The new behavior provides the SAF user the control to clear a building using user specified, server specified, or a mixture of inputs. The design for this behavior came from extensive research performed in official U.S. Army field manuals (FMs).

The template for how the Clear Building behavior operates is as follows:

1) The user selects the unit (squad) and the task frame "Clear Building". Required as part of defining the task, he then specifies an initial room to clear and a stack point outside the building where the squad will gather prior to beginning the assault. If the user does not wish to use a server, he is provided the option to specify the order in which to clear the

rooms.

2) Once the squad gathers outside the building at the stack point, a fire-team is selected and approaches the first room. They will then stack outside the nearest doorway to that room and wait while gathering intelligence on the room to clear. The amount of time is a function of the sensor they have been configured to use and is the time needed to gather SA data about the contents of the room. Once the time has elapsed, the unit performs the actions to enter the room that the server has specified (e.g., charge in, abort, throw grenade, etc.) with the fire permissions that the server has specified.
3) The squad will repeat the above sequence until one fire-team has successfully entered and cleared the initial room. Once the initial room has been secured, the squad leader will enter the room and use it as a command post.
4) Now that the initial entry is accomplished, the fire-teams within the squad will begin a leapfrog technique of moving through the building. The SAF will send the server a list of all the spotted doors in the current room, as well as a status (cleared or not) for each door. The server will then respond with a door to enter, the fire-team to go through the door, and how long they should gather SA data in the stacking location outside the door.
5) Once the above information has been received, the SAF sends the server intelligence on the room's contents, and the server returns the room entry action to perform, the fire permissions to use upon entering the room, and the fire-team to perform the action. With this information in hand, the current fire-team will perform the indicated actions, and if those actions were to enter, they will attempt to secure the room.
6) The above steps will repeat until every door that has been spotted has been entered and the attached room cleared. Once all accessible interior doors (going outside and re-entering is not dealt with) have been cleared, the task is at an end and that building is considered secured.

In general, this behavior follows the doctrine set out in the FMs mentioned previously. Certain "real" aspects of building clearing were not modeled, primarily due to the limitations of the SAF. Examples of this are that instead of using doorways explosives are usually used to create an entry, and a member of the squad would typically remain behind in each room to maintain security.

3.4.3 Demonstrate Behaviors within a SAF

Demonstration of a SAF with a server-supported behavior will show the rapid ability to modify behavior models and have those changes immediately show up in entities within a SAF. Once an entity is placed on the synthetic battlefield, the user will have the ability to select whether it should continue to use the standard SAF behavior attributes or if the attributes should be obtained from a high fidelity model resident on an external server. The model resident on the server may be easily modifying and this new version of the model will immediately be used by the SAF. This is a demonstration of the power, flexibility and usability of the client-server architecture for providing high fidelity behavior representation.

Shown in Figure 3-9 is a snapshot of an infantry squad in a building performing a "Clear Building" operation. DISAF is communicating via DIS to a behavior server that is simulating the decision making process of the squad leader of a fire team. This process includes a

subscription of the Dismounted Infantry squad to a server that can provide the Clear Building service information being requested. Once the subscription process has been completed, the squad passes building and situational awareness information (e.g., number of spotted enemy, friendly, and neutral units) to the server it is subscribed to. The server then takes the provided information, and returns the time to wait outside a room gathering intelligence, the door to enter a room through, the fire permission for the unit, and the action to take to enter that doorway. Possible actions include throwing in fragmentary grenade, a smoke grenade, or flash-bang grenades, charging in, and aborting the mission.

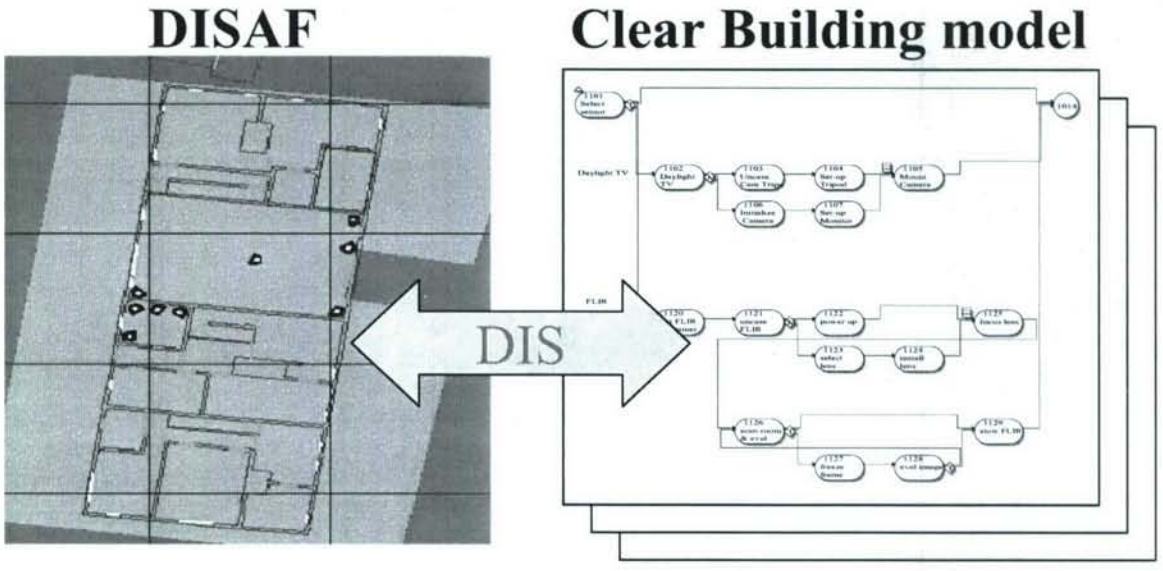

Figure 3-9. Clear Building in C-S architecture.

This demonstration of the use of the client-server architecture illustrates the process in which an external model can interact and influence decisions and behaviors taking place with the CGF. Furthermore, simply by modifying the model (or replacing it altogether), that is, without doing any modification to the SAF, a model within the server can be altered and the effects can immediately be included within the SAF.

3.4.4 I/O

Within the SAFs are behaviors, or "hooks". They are the effects that drive entities to behave and react to the environment. A SAF such as DISAF contains libraries that effect behaviors of entities. Each library has hooks for behaviors that can be modeled external to the SAF within a server. Two libraries have been added that allow these behaviors to be affected by the server. The two libraries are libsubscriptionmgr, that is the subscription manager, and libdatahandler that handles the transfer of data between the SAF and the server.

The inclusion of these two libraries along with the modifications to the behavior libraries will provide for the usage of composable behaviors while also maintaining the capabilities previously present within the SAF. The SAF user will have the ability to select which entities should attempt to use a server. Only if the user makes such a selection and the appropriate behavior (or service) is available will it be used.

3.4.5 Clear Building scenario

To demonstrate the new Clear Building capabilities, a simple scenario was developed. The scenario components consist of an infantry squad (blue entities) assaulting the building, three armed enemy combatants (red entities) defending the building, and a civilian (green entity) non-combatant. The scenario is created by having a US IC Squad perform the task of "Squad Clear Building", with a defined initial room to clear and a squad gathering point exterior to the building. The illustrations in Figure 3-10, Figure 3-11, Figure 3-12, and Figure 3-13 show the sequence of events as they perform this task. This example scenario consists of the clear building behavior including:

1. approaching the building,
2. stacking and engaging enemy in the first room,
3. securing the first room and having the squad leader enter, and finally
4. both fireteams coordinating the clearing of all the rooms and the enemy neutralized.

The result in this example is the clearing of all the rooms of the building, all three enemies and one friendly dead, and the civilian taken unharmed.

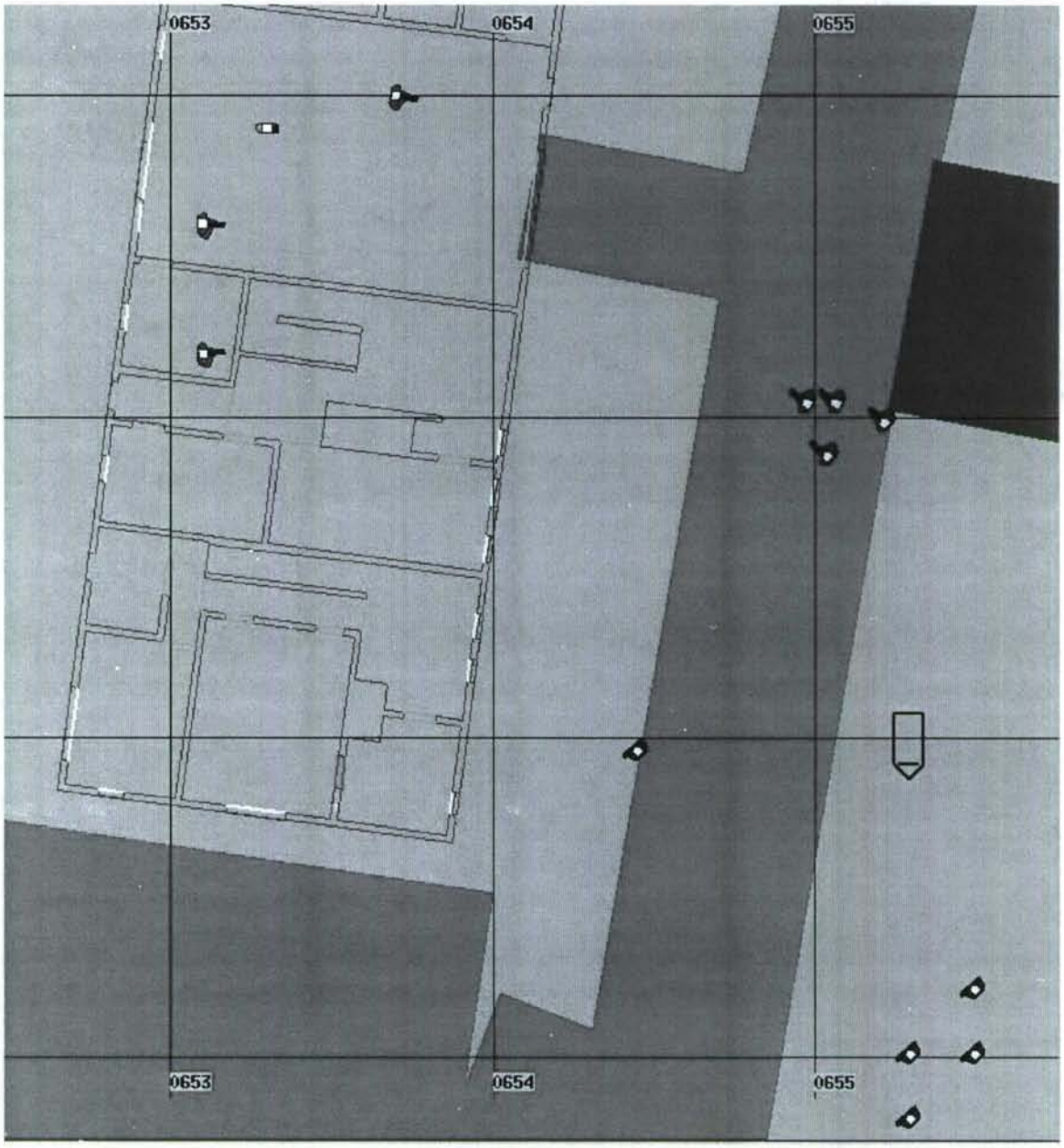

Figure 3-10. Preparing to enter the building.

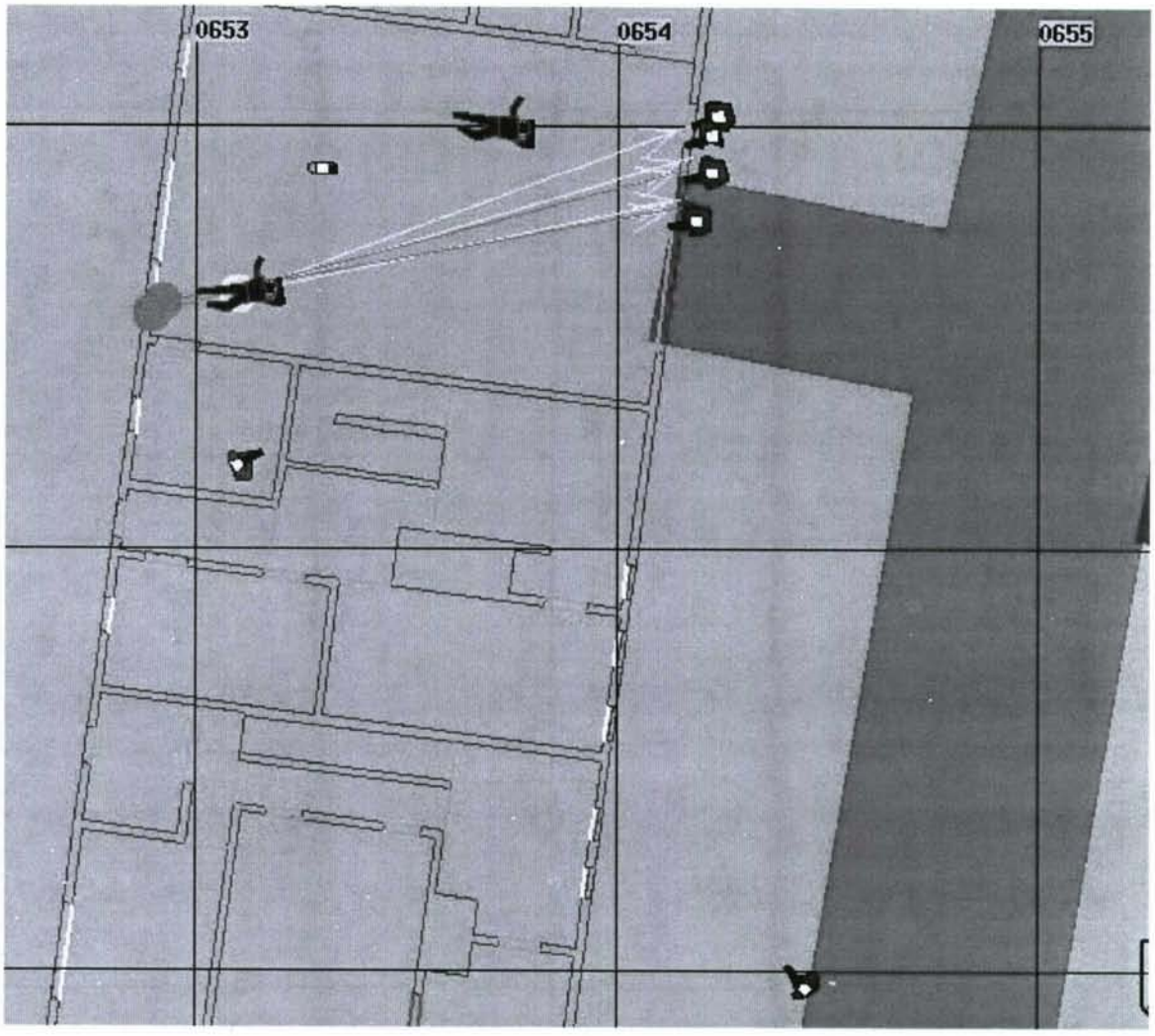

Figure 3-11. Assaulting the entry room.

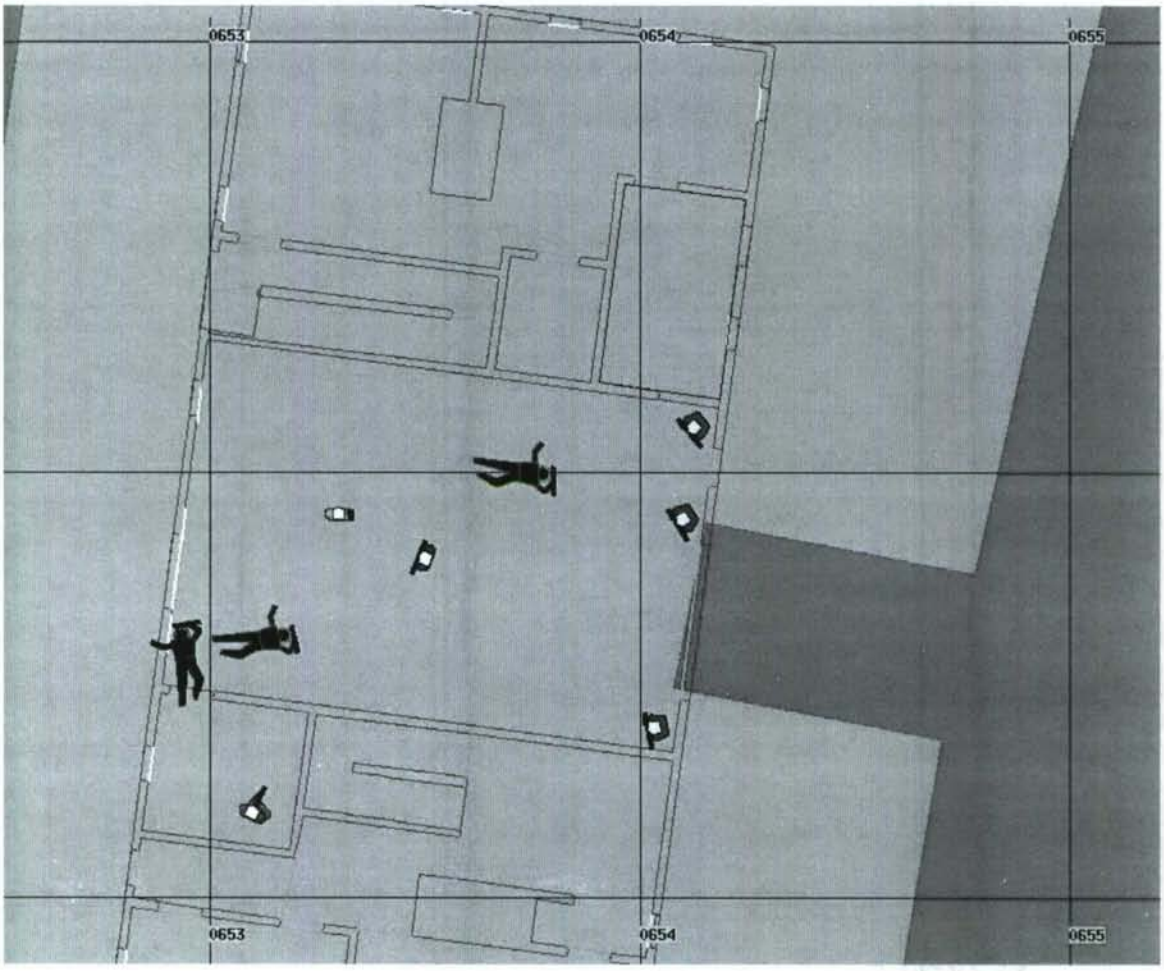

Figure 3-12. First room secured, squad leader inside.

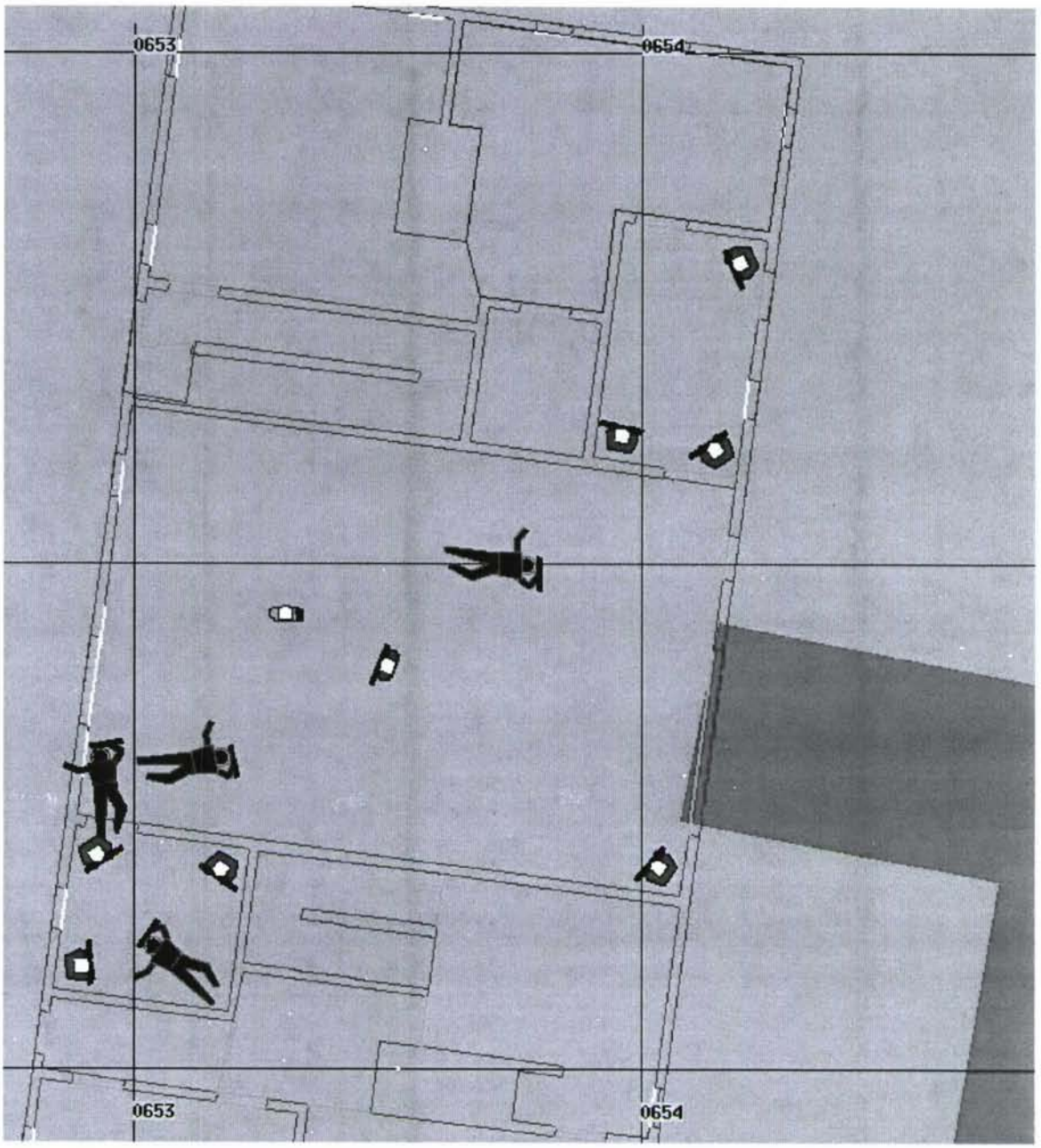

Figure 3-13. Entry rooms secured, all enemy dead.

3.5 Client-Server Architecture

Figure 3-14 shows DISAF acting as the client and communicating with a server via a DIS network. A subscription manager in the SAF communicates using SIMAN interactions with a "server-side subscription manger" that resides in server's middleware. Similarly, a library in the SAF and a translator in the middleware processes and distributes data. The middleware communicates to the simulation engine, that is, Micro Saint, via a Common Object Module (COM) interface. Notice that the client (DISAF in this example) runs under the Linux operating system and the server components (i.e., Middleware, Micro Saint, RPD) run under the Windows operating system, and they intercommunicate using DIS.

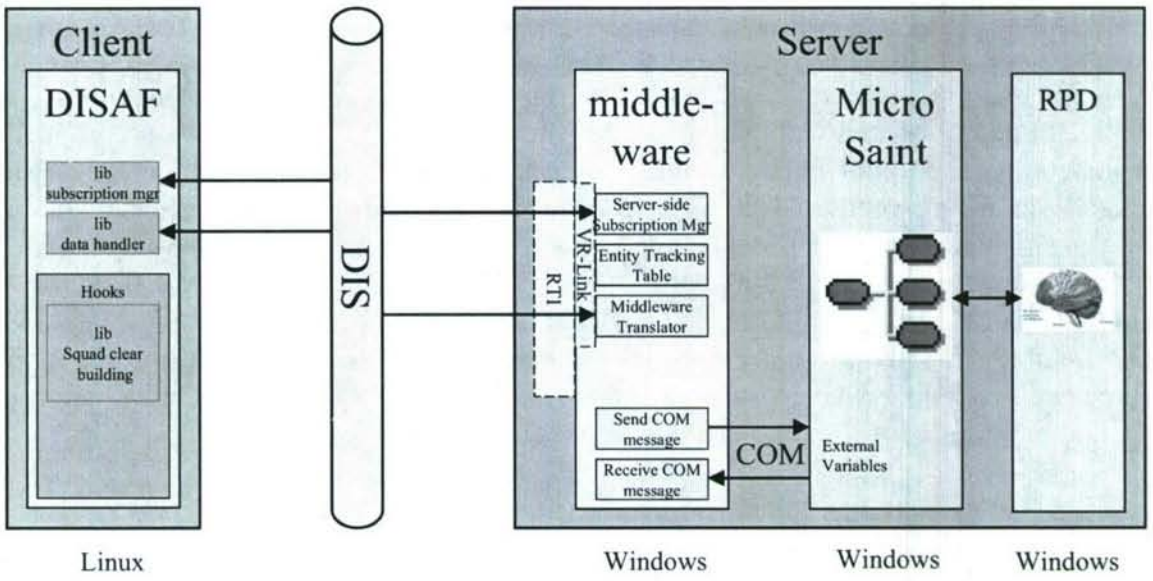

Figure 3-14. Client-Server Architecture for Squad Clear Building.

Figure 3-15 shows how this architecture is scalable to accommodate multiple SAFs and multiple servers. The subscription process provides the needed functionality for load balancing and allows simultaneous capability for multiple behaviors to be served by multiple server models. In addition, the server (Micro Saint in this particular application) can be augmented by a cognitive processing capability such as RPD. Using this architecture provides the capability to have as many squads as desired, all performing clear building operations, and all using high fidelity behavior models. The behavior model developed will accurately and realistically represent their behaviors including cognitive processing. In addition, by having the behavior models external to the SAF, modifications to the algorithms can be done easily and quickly.

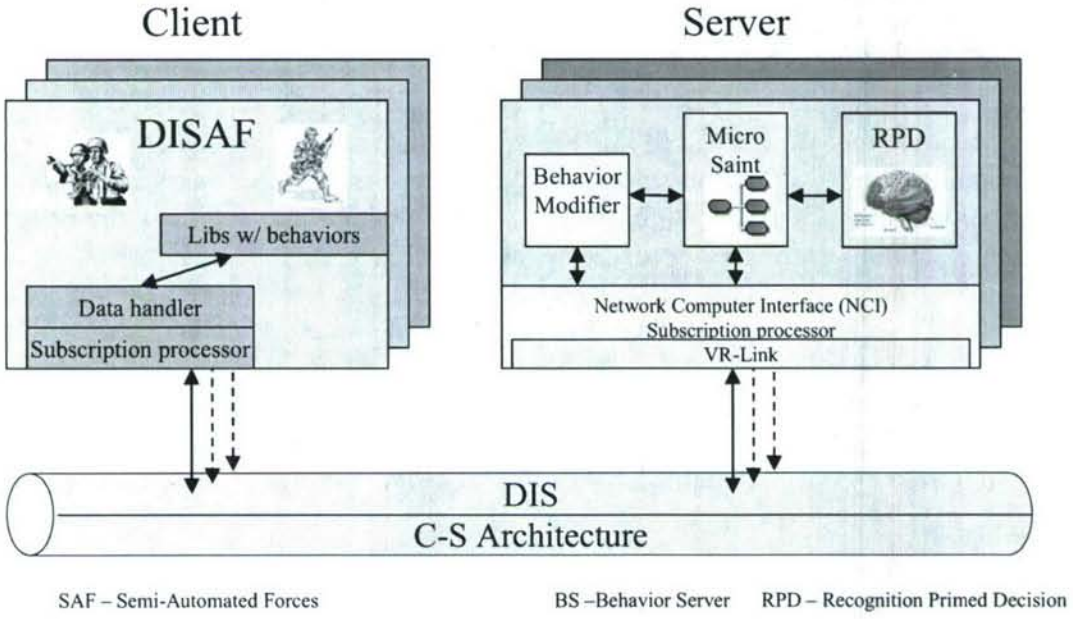

Figure 3-15. Scalable Client-Server Architecture.

4.0 Results

Through this program we were able to implement improved behavioral representation for Dismounted Infantry entities within both DISAF version 7.1 and OTB-JVB. We accomplished this using two types of human behavioral models; a task network model that allows us to make some decisions and also determine human performance parameters, and a cognitive model that allow us to simulate human decision making. In our approach, we communicate situational awareness information from the SAF to the Human Behavior Model (HBM). We also provide decisions and behaviors to the SAF from the HBM. In our approach, the behaviors that we can provide are non-generic, much more doctrinally correct and realistic than those residing in the baseline version of either SAF. No longer is there a need to use and be restricted by hard coded and limited behaviors.

In the accomplishment of this project, we did encounter some issues and generated some lessons learned. These are as follows:

- Having a flexible statement of work allowed us to redirect and refocus our efforts to respond to our customer. The initial focus of having many HBR laboratories interact quickly became cumbersome and was inhibiting to all of the DMSO HBR Lab programs. In this program we were able to redirect our efforts to DI entities in a MOUT environment. In the end the customer's needs were met and they received a high quality product that provides significant value to the Government.
- Obtaining CGF applications and updates was time consuming. We, along with DMSO, spent a considerable amount of time obtaining versions of CGF applications from STRICOM, even with DMSO assistance. We eventually did obtain updated SAF applications and were able to successfully integrate our improved MOUT behaviors into two versions of SAFs, but with some limitations. If we could have received these applications sooner, we could have been more successful in the accomplishment of our integration objective. The lesson here is that DISAF and OTB-JVB applications should be easier to obtain, or at some point in a program the CGF application version should be frozen.
- The last lesson that we learned in this program is that as we increase behavioral representation for a CGF application, we encountered modeling fidelity limitations in other parts of the CGF application. When we included our higher fidelity behaviors, we encountered more and more anomalies within the CGF application. In certain instances, the building structures within the terrain database were not correct, for example windows were coded as doors. Also, because we were implementing a more doctrinally correct building clearing behaviors, the routes and paths that entities take is a key aspect of the behavior. The current routing algorithms within the SAFs are not robust enough for this. The routing algorithms do not take into account that routes in buildings are different than in open terrain and need to consider things such as interior and exterior doorways.

5.0 Conclusion

While this HBR Lab program was directed at DI entities conducting MOUT operations, the program has implications and benefits far beyond this domain. Our enhancements can be applied to many different entities within SAFs. Our enhancements and this approach could be

easily applied to other CGF applications. Our incorporation of heterogeneous HBMs for our HBR server demonstrates the flexibility and robustness of our approach. We believe that this effort provided a significant advancement of the state of the art of human behavioral representation for modeling and simulation.

6.0 References

Anderson, J. R. and C. Lebiere (1998). The Atomic Components of Thought. Mahwah, NJ, Lawrence Erlbaum Associates.

Hintzman, D. L. (1984). "MINERVA 2: A simulation model of human memory." Behavior Research Methods, Instruments & Computers 16: 96-101.

Hintzman, D. L. (1986). Judgments of Frequency and Recognition Memory in a Multiple-Trace Memory Model. Eugene, OR, Institute of Cognitive and Decision Sciences: 1-94.

Hintzman, D. L. (1986). ""Schema Abstraction" in a Multiple-Trace Memory Model." Psychological Review 93(4): 411-428.

Klein, G. (1993). A recognition-primed decision (RPD) model of rapid decision making. Decision Making in Action: Models and Methods. G. Klein, J. Orasanu, R. Calderwood and C. E. Zsambok. Norwood, NJ, Ablex Publishing Corp.: 138-147.

Klein, G. (1998). Sources of Power: How People Make Decisions. Cambridge, MA, The MIT Press.

Klein, G., R. Calderwood, et al. (1985). Rapid decision making on the fire ground. Yellow Springs, OH, Klein Associates, Inc.

LaVine, N., Peters, S., Napravnik, L. (2002). Improved Behavioral Representation for Operations Other Than War Within Aggregate Level Simulations – Final Report. Boulder, CO, Micro Analysis & Design.

Pearl, J. (1988). Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. San Francisco, Morgan Kaufmann Publishers, Inc.

Pearl, J. (2000). Causality: Models, Reasoning and Inference. Cambridge, Cambridge University Press.

Peters, S. LaVine, N., Napravnik, L. (2002). Composable Behaviors in an Entity Based Simulation. Boulder, CO, Micro Analysis & Design.

US Army. FM 3-06.11 "Combined Arms Operations in Urban Terrain"

US Army. FM 7-8 "Infantry Rifle Platoon and Squad",

US Army. TC90-1 "Training for Urban Operations",

US Army. FM 7-10 "Urban Operations"

APPENDIX A

Squad Clear Building Model

Overview

Dismounted Infantry (DI) personnel involved in operations to clear buildings are represented in Dismounted Infantry Semi-Automated Forces (DISAF) and OneSAF Testbed Baseline (OTB) Joint Virtual Battlespace (JVB) using models that focus on the actions soldiers perform. At best these models provide a cursory representation of the decision making process driving these actions. Currently untreated, the qualitative and sometimes illogical processes of the human mind are a pivotal component that injects the realism and variability required for these models to fulfill their intended purpose.

To demonstrate and document the actions and decisions made by soldiers while clearing a building, Micro Analysis and Design created a task network model incorporating human characteristics. This model focuses on cues influential in the soldier decision making processes and combines cues in a human like manner based less upon distinct quantitative differences and more on perceived differences (e.g. how many adversaries a soldier discerns rather than all adversaries including those unseen). The model also incorporates various random influences that represent a degree of "gut feeling." This model attends to nuances that distinguish soldiers from omniscient robots that strictly follow military techniques, tactics, and procedures, hence allowing room for human error and variance. The relationship between actual and perceived reality is not always perfect.

Incorporating realistic opportunities for soldier error and misperception, the model entails a squad of dismounted soldiers approaching a building from a remote location as well as the core processes of entering the building, and clearing each room. Building exit strategies are not covered nor are details of engagement outside the building. Such modeling efforts would require additional topographic and geographic details that could be implemented in the future. The tasks and procedures contained in this model are the product of analyzing Field Manuals (FM) and visualization of the building clearing process.

Built in Micro Saint, a task network modeling tool ideal for modeling human behavior, each top-level network represents a supporting objective (see Figure A-1). Task network modeling provides an easy way to organize sets of tasks (e.g. entering a door or engaging an enemy) along with decision logic that determines which tasks are performed. The model is arranged in a branching flow chart representing the flow of tasks, such as an infantry unit moving through a building. Depending on conditions generated within the model (e.g. aggressive, well armed enemy soldiers as opposed to unarmed or passive ones), entities in the model react differently and choose different courses of action, i.e. paths. The basic task network model is comprised of a network of paths linking tasks. Decision nodes handle logic when a task is succeeded by more than one task. Each task has an associated execution time, release condition, beginning and ending effects. Task times are based on actions squads must perform and conditions they encounter. For example, in traveling to a door, time is calculated as a function of distance to the door and speed of travel. Release conditions are conditions that must be true for a task to begin. For example, if a DI squad has less than three members in good health, they must wait until they have reinforcements before they can begin the task of selecting the next room to clear.

Beginning and ending effects are comprised of actions that occur in the event. If the task is engaging the enemy, beginning and ending effects contain logic that determines the entity's success in killing adversaries, casualties the entity endures, ammunition expended by each side, and decision logic for continuing the fight or retreating to regroup. In this manner the model generates realistic scenarios and squads respond according to cues they encounter.

Clear Building Model

Micro Analysis and Design created a task network model of a squad of soldiers clearing a group of buildings in Micro Saint. This model is self-contained in that it not only models the actions of the soldiers as they move from building to building and from room to room engaging enemy and civilians, but it also creates a small "city" of buildings with room and enemy and civilian personnel with personality profiles, some carrying weapons. Execution of the model creates a city and populates it, sends the DI squad in to clear the buildings, and saves results of the operation in files for later analysis.

The task network model begins with an initialization process that creates a fictitious city of between one and ten buildings. These buildings are described by various characteristics such as:

- Number of rooms
- Number of doors to each room
- Door status as open, closed, or fortified
- Number of enemy soldiers and civilians in each room
- Adversary armaments

It also addresses internal fortifications for each room and the DI's fire permission with regard to each building. Once the city is populated, a DI squad is placed in the city at some random distance from each building. This brings us to the first set of tasks in the first network of the model, Select the Building to Clear (*Sel Bld To Clear* in Figure A-1).

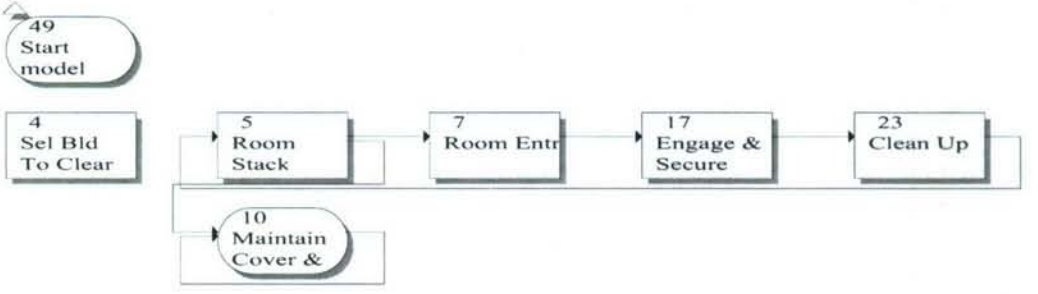

Figure A-1. Clear Building - Top Level Network.

Tasks that compose the first network are shown in Figure A-2. In the first task, *Select Building* (task #1), the DI squad uses distance to the buildings as well as each building's priority to clear, path concealment from squad location, and terrain difficulty to decide which building to approach. Upon selecting a stack point outside the building entrance of choice, the squad proceeds to the building in a posture and speed dependent upon the priority to clear this building and the current environmental conditions. For example, if the priority to clear a building is urgent and the distance to the building is minimal, the squad will likely move quickly (i.e., run). The model determines the time it takes a squad to *Approach in Stealth* (task #3 of Network 4, see Figure A-2) via the distance to the building and the speed determined in the *Plan Approach* (task #2) task.

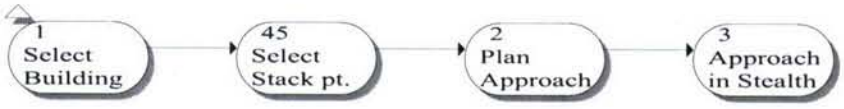

Figure A-2. Network 4: Select Building To Clear.

Once at the building entrance, DI squad actions become generic to entering a standard door of a room. Having selected a door to enter, the unit performs *Select Sensor* (task 11) of Figure A-3. The model also initiates task 10, *Maintaining Cover and Concealment* with this action being a parallel task performed in the periphery while they focus on scanning and planning. Maintaining cover and concealment rises to the forefront in room entry and engagement maneuvers. While uncleared rooms remain, the DI squad continues to loop back to *Select Room* (task 9) after clearing a room and moving civilians and prisoners.

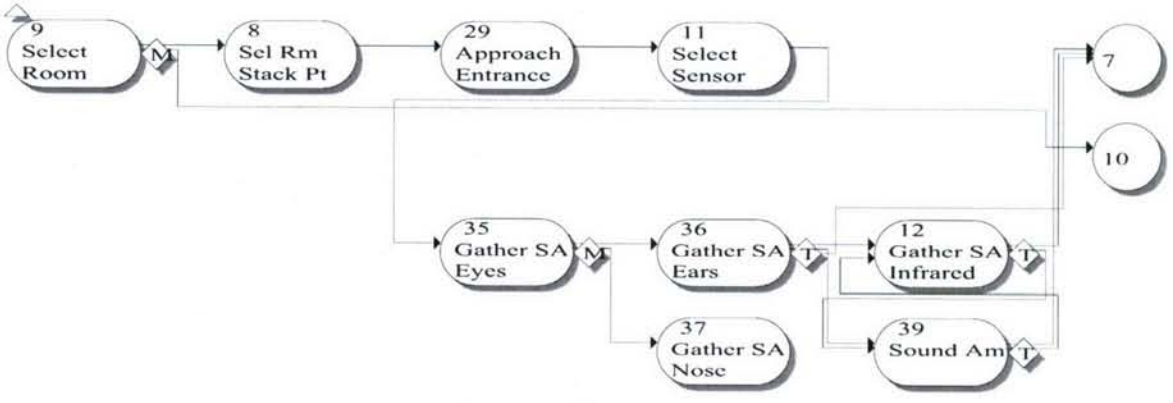

Figure A-3. Network 5: Select Stack Point and Gather Situational Awareness.

Squads select rooms based mainly on perceived proximity (task 9). They never want to leave potentially occupied rooms behind them due to the potential for an ambush. They choose the next room by finding those rooms nearest to them and choosing the door to enter (if more than one per room) by observing the door status. Open doors are the highest priority as they indicate enemy presence and are easier to inspect. Fortified doors indicate enemy presence as well. Once the squad selects a room, they approach it in task 29, *Approach Entrance*, at a designated speed. Approach time is calculated based on distance to the chosen door. At this point in the model taxonomies for clearing a room and entering a building resolve into one procedurally uniform protocol.

DI squads deploy with sensors ranging from normal vision and hearing to sound amplifiers and infrared sensors that more accurately depict room contents. Usage of these sensors increases the squad's chance of forming an accurate picture of the room before entering. At the building entrance, soldiers are more likely to take more time and scan carefully. Their awareness may be lower prior to experiencing the general interior environment; they can establish a feel for the building with a thorough scan. If the team has an infrared sensor it will almost always use it instead of sound devices. The sound amplifier is useful exclusively in environments with less than 75 decibels (dB) of noise. Greater noise interferes with one's ability to distinguish between background noise and movement. After scanning a room the squad is ready to proceed with entering via network 7, *Room Entry* (see Figure A-4).

Figure A-4. Network 7: Room Entry.

Based on the fire permission assigned to a building as well as the perceived number of enemies in a given room, the DI squad will choose either an aggressive entry plan to preemptively quash resistance, or a more cautious route to preserve civilian lives (see task 13, *Room Entry Plan*). If the door is fortified they will use a C4 charge to blow it. If the door is open and they perceive enemies inside but few civilians, they will throw a grenade as the open status of the door provides additional situational awareness (SA) in terms of enemy and civilian location; they might be able to toss a grenade in with a low possibility of civilian harm. If there are many civilians or fire permission is not free, the squad will at most use M16 fire to enter the room and kill adversaries with a lower probability of civilian casualties. Alternate methods entail tear gas, and commanding the enemy to stand down. As in other approach tasks, the duration of *Move to Entrance* (task 14) is calculated via a calculated rate of movement and distance to the entrance. *Assume Formation* (task 30) will become functional as more information regarding possible squad formations becomes available. Soldiers apply the selected room entry method in *Apply Method of Entry* (task 15). Adversaries may be killed depending on the entry method applied and the chance they are within proximity of a grenade or C4 blast. Entering the room leads to network 17, *Engage and Secure Room* (see Figure A-5).

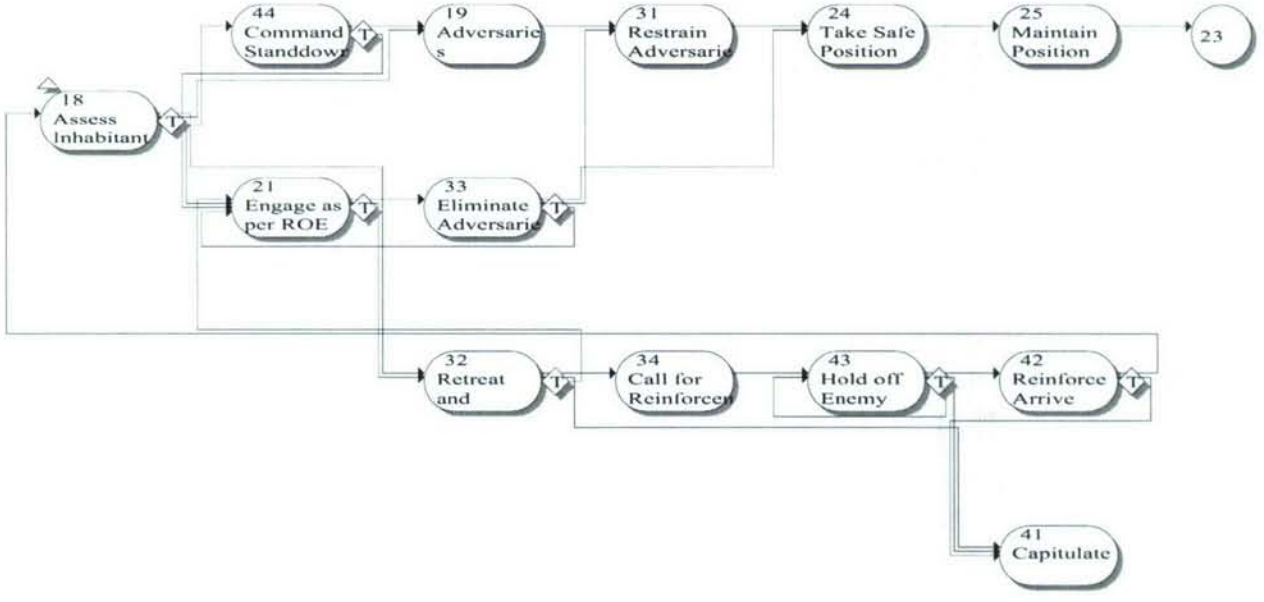

Figure A-5. Network 17: Engage and Secure Room.

Upon entering a room the DI squad assesses the situation in terms of the potential for squad casualties, adversary fortifications, aggressiveness, and whether adversaries are attacking back. The squad attempts to distinguish civilians from enemy soldiers as well as briefly analyzing enemy armaments. If the enemy

is dangerous enough or if squad sensing efforts have led them to greatly underestimate the force in the room, they *Retreat and Regroup* via task 32 (see Figure 2-1). If adversaries are passive and do not attack or the room is devoid of enemy soldiers, room inhabitants travel the path to surrendering directly, *Adversaries Surrender*, or the squad executes a *Command Stand Down*. If room inhabitants attack at all the DI squad will either engage them or retreat and regroup. This network has a great deal of flexibility to accommodate changes in picture. If enemies feign surrender the DI squad engages. The squad can also retreat and re-enter a room if the battle momentum changes or if regrouping is needed. If a squad takes casualties of one third (33%) while adversaries have sustained less damage, the squad must call for reinforcements as they are not making sufficient headway. After calling for reinforcements the squad must hold off the enemy until the reinforcements have had time to travel the distance to the building and reach the room they are attempting to clear. If reinforcements do not arrive before adversaries force concession, the DI squad will capitulate. If reinforcements are on time, the augmented squad will assess the picture as a new force.

Engage as per ROE (task 21), is in a loop with *Eliminate Adversaries* (task 33). These two tasks represent the squad engaging and pulling back to engage again. Each time this loop is performed the squad engages the enemy and takes tolls according to the potential for casualties established in task 18, *Assess Adversaries* (see Figure A-5). Both sides also expend bullets in a manner consistent with a difficulty level based on fortifications and danger. Some adversaries may surrender before their teams are eliminated. Others may require a fight to the end. In either case, the end of a victorious battle leads to *Clean Up* detail in network 23 (see Figure A-6).

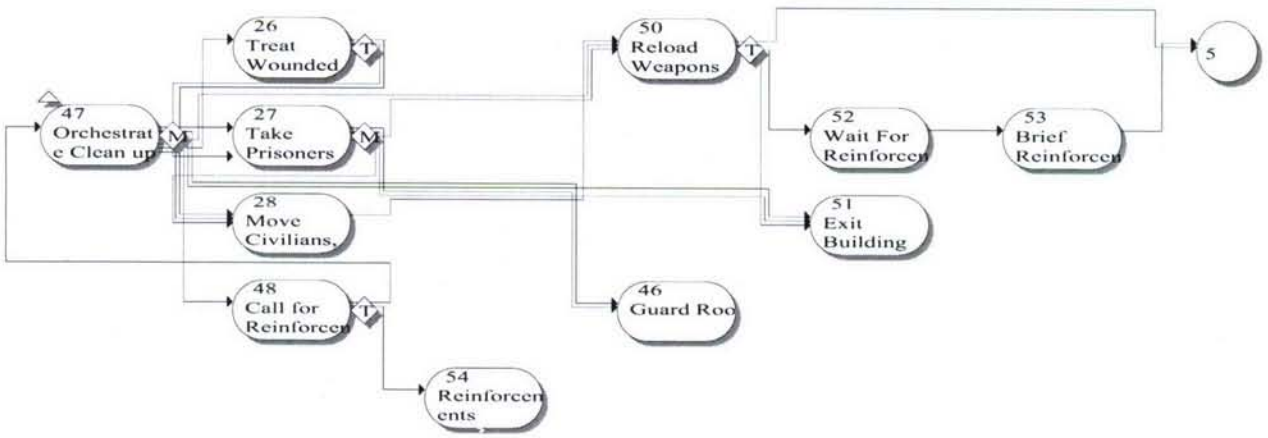

Figure A-6. Network 23: Clean Up.

Task 47, *Orchestrate & Clean up*, is the central decision making task. If less than three (3) members of the squad are able-bodied, they call for reinforcements. One of the soldiers will travel to the building entrance to meet reinforcements if more than one is in active condition (one soldier must remain in the room as a guard). The presence or lack of a guide to the room influences the time it takes reinforcements to arrive. If there are one or more able-bodied soldiers in the room, they will proceed to treat the wounded, take prisoners, move civilians and reload their weapons until help arrives. They will then brief the reinforcements and proceed with the next room. Each task in this network executes based on conditions. Decisions and paths allow the squad to perform only those tasks that apply (e.g. if there are no civilians, we don't move any).

Task 51, *Exit Building*, executes upon clearing all rooms or shortage of ammunition. In either occurrence the DI squad exits the building at a designated rate of speed. The distance to the entrance is

tracked with squad depth into the building. Since all rooms behind them are cleared the squad exits from whence they came and without resistance.

Conclusion

With functionality to simulate human-like courses of action, entities simulated in this model make qualitative, human-like decisions on a mixture of logical and "gut-feeling" inputs. With decisions based on perceived information as opposed to cold, hard facts, squads in the model make decisions based on what they have the opportunity to know rather than what they should know in an ideal world. Built using Micro Saint flexibility and functionality, the model is robust to handle multiple squads clearing buildings in a simulated city environment. Potential fidelity of future Micro Saint models that interact with the SAF could dramatically increase realism hence usefulness of SAF models in predicting human response to scenarios. This insight could prove useful in training soldiers for proper response and building clearing operations in a MOUT environment.